DAGGER: Data AuGmentation GEneRative Framework for Time-Series Data in Data-Driven Smart Manufacturing Systems

David Blank[1], Daniel Ospina Acero[2], Nicholas Hemleben[3], Andrew VanFossen[4], Frank Zahiri[5], Mrinal Kumar[6]

[1,2,4,6] *Mechanical & Aerospace Engineering, The Ohio State University, 201 W. 19th Avenue, Columbus, OH 43210, USA*
blank.136@buckeyemail.osu.edu
ospinaacero.1@osu.edu
vanfossen.24@buckeyemail.osu.edu
kumar.672@osu.edu

[3] *PointPro Inc., 7658 Johntimm Ct, Dublin, OH 43017, USA*
nicholas@pointprousa.com

[5] *402 Commodities Maintenance Group (CMXG), Warner Robins Air Logistics Complex, Robins AFB, GA 31098*
feraidoon.zahiri@us.af.mil

ABSTRACT

Performance of digital twins (DTs) in smart manufacturing is heavily data dependent, especially when physics-based computational models are not available or difficult to obtain in practice, as it's the case in most modern manufacturing scenarios. However, in manufacturing applications the availability of data is often limited and involves high dimensional signals. In this work we present the Data AuGmentation GEneRative (DAGGER) framework, which is a deep-learning-based tool combining the strengths of autoencoders (AEs) and generative adversarial networks (GANs) to robustly, efficiently and reliably augment the available sensor data to train the data-based computational models of DTs for smart manufacturing. The DAGGER framework uses the learned latent space from an AE into the training process of the generator in a GAN. This provides increased stability in the convergence of the GAN's discriminator/critic when working with very small *training* data sets, and helps the GAN's generator to more accurately and robustly capture the structure in the sensor data. We corroborate the efficacy of the DAGGER framework in two ways, one in which we directly contrast the synthetically generated time series samples with real ones from publicly available sensor data in a manufacturing application (using performance metrics based on the similarity of mean signals, variance signals, KL divergence, signals in Fourier domain, auto-correlation signals, etc.), and one in which we evaluate the adequacy of the synthetically generated time series samples to perform system identification of a black box dynamic system. In all the cases we compare the results from the DAGGER with those from a Riemannian Hamiltonian Variational AE (RHVAE). We show that the DAGGER's performance is in general satisfactory, and comparable with that of the RHVAE in all the considered evaluation scenarios.

1. INTRODUCTION

As industries transition into the Industry 4.0 paradigm and increasingly incorporate smart manufacturing practices in their operation, the relevance and interest in concepts like DT are at an all-time high. DTs offer direct avenues for industries to make more accurate predictions, rational decisions, and informed plans, ultimately reducing costs, increasing performance and productivity. The adequate operation of DTs in the context of smart manufacturing relies on an evolving dataset relating to the real-life object or process, and a means of dynamically updating the computational model to better conform to the data (Wright & Davidson, 2020). This reliance on data is made more explicit when physics-based computational models are not available or difficult to obtain in practice, as it is the case in most modern manufacturing scenarios. For data-based model surrogates to "adequately" represent the underlying physics, the number of training data points must keep pace with the number of degrees of freedom in the model, which can be on the order of thousands. However, in niche industrial scenarios like those in manufacturing applications, the availability of data tend to be limited (on the order of a few hundred data points, at best) (Diez-Olivan, Del Ser, Galar, & Sierra, 2019), mainly because a manual measuring process typically must take place for a few of the relevant quantities, e.g., level of wear of a tool. In other words, notwithstanding the popular notion of big-data, there is still a stark shortage of ground-truth data when examining, for instance, a complex system's path to failure. In this work, we present a framework to alleviate this problem via modern machine learning tools, where we show a robust, efficient and reliable pathway to augment the available data to train the data-based computational models.

Small sample size data is a key limitation in performance in machine learning, in particular with very high dimensional data (Goodfellow, 2016). Current efforts for synthetic data generation typically involve either Generative Adversarial Networks (GANs) or Variational Autoencoders (VAEs) (Figueira & Vaz, 2022; Demir, Mincev, Kok, & Paterakis, 2021), and both have been used in numerous applications, including those with very high dimensionality, e.g., 3D MRI images. However, there remain hurdles to overcome in the context of smart manufacturing:

- Paradoxically, these methods fall under the umbrella of deep learning and therefore themselves require vast numbers of hyperparameters to optimize (Hutter, Lücke, & Schmidt-Thieme, 2015). Traditional GAN and AE architectures therefore require large data sets for adequate training. Thus, new network architectures that can be effectively trained with very small data sets (e.g., a few hundreds of data points) need to be developed.
- Most applications of GANs and AEs target image analysis and synthesis (Smith & Smith, 2020; Doersch, 2021), which renders traditional network architectures poorly suited for time-series data generation. Although there has been recent work on augmentation techniques for time series data with GANS and/or VAEs (Iglesias, Talavera, González-Prieto, Mozo, & Gómez-Canaval, 2023; Yang, Li, & Zhou, 2023), the problem remains large open, especially regarding the efficiency of both the training and the generation operations. Consequently, robust and efficient architectures tailored to time-series sensor data generation must be defined.
- Standalone GAN or VAE architectures inherently bring along a few difficulties. For example, GAN models are susceptible to mode collapse, training instability, and high computational costs when used for high dimensional data creation (Khanuja & Agarkar, 2023, Ch. 13). On the other hand, the encoding of VAEs greatly reduces dimensional complexity of data and can effectively regularize the latent space, but often produces poor representational synthetic samples (Shao et al., 2020). Thus, alternative neural network architectures that alleviate such practical issues need to devised.

In light of these challenges, here we present the Data AuGmentation GEneRative (DAGGER) framework, which corresponds to a hybrid AE+GAN architecture specifically conceived to generate synthetic high dimensional time series data when the training data sets are very small. The DAGGER scheme uses the learned latent space from an AE in the training process of the generator of a GAN, increasing robustness and stability in both training and synthetic sample generation.

2. PROBLEM STATEMENT

A data set $\mathcal{S}$ contains K time series signals that come from a sensor in a manufacturing device (e.g., a vibration sensor placed at the spindle on a milling machine). The k-th element in $\mathcal{S}$ is the time series signal $\mathbf{x}_k$ containing N time samples; i.e., $\mathbf{x}_k = [x_k(t_1), x_k(t_2), \ldots, x_k(t_N)]$. Our objective is to obtain new time series signals that "resemble" those in $\mathcal{S}$. More concretely, if we interpret every $\mathbf{x}_k$ as a sample drawn from a probability distribution $p_{\text{data}}(\mathbf{X})$, in principle we simply wish draw new samples from $p_{\text{data}}(\mathbf{X})$. The problem, however, is that $p_{\text{data}}(\mathbf{X})$ is unknown, and we have to first devise a model for it.

To that end, let us define a probability distribution $p_{\text{model}}(\mathbf{X};\boldsymbol{\theta})$ that approximates $p_{\text{data}}(\mathbf{X})$, with $\boldsymbol{\theta}$ representing the set of defining parameters of p_{model}. With a slight abuse of notation, our first step is thus defined by the following optimization problem (Goodfellow, 2016):

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{D}\big(p_{\text{data}}(\{\mathbf{X} = \mathbf{x}_k\}_{k=1}^{K}), \\ p_{\text{model}}(\{\mathbf{X} = \mathbf{x}_k\}_{k=1}^{K};\boldsymbol{\theta})\big) \ , \qquad (1)$$

where $\mathcal{D}(\cdot,\cdot)$ represents some measure of distance between two probability distributions (e.g., Kullback-Leibler divergence). Once $\boldsymbol{\theta}^*$ is found, we obtain a new time series signal $\hat{\mathbf{x}}$ by sampling from $p_{\text{model}}(\mathbf{X};\boldsymbol{\theta}^*)$.

Now, for manufacturing systems, the number of time series signals available for "training" is typically very small (in the order a few hundreds), and the number of time samples in a time series is large (in the order of several thousands), i.e., $N \gg K$. Consequently, the problem in (1) must be solved under those constraints: high-dimensional data with very few samples for "training". Here we present a robust framework to address these challenges.

3. AE+GAN ARCHITECTURE: DAGGER FRAMEWORK FOR SYNTHETIC TIME SERIES GENERATION

In this section, we describe the methodology developed to solve the problem described in the previous section. The methdology is dubbed the DAGGER framework. The DAGGER corresponds to the combination of two different deep neural network architectures working in tandem to produce synthetic samples of time series signals under the constraints of high dimensionality and very few samples for training, which are typical in manufacturing scenarios. The two network architectures defining the DAGGER framework are an

AE network and a GAN, the former dealing with the transformation of the original data to and from a lower dimensional space, and the latter dealing with the generation of the synthetic samples. A schematic view of the DAGGER framework is shown in Figure 1.

Following the process illustrated in Figure 1, first the (trained) AE network of the DAGGER takes the original time series data (i.e., set $\mathcal{S}$) and transforms it into a considerably lower dimensional space via its encoder network. The output of the encoder is then fed to the critic network of the GAN, along with synthetic samples produced by the generator network of the GAN, and the critic simply provides a rating on how real it considers them to be. The performance of the critic when differentiating real samples from synthetically generated ones is measured, and this information is then used to update the defining parameters of both the critic and the generator networks of the GAN. This process is repeated until an acceptable level of performance is reached (when the synthetic samples are indistinguishable from the real ones to the critic). Lastly, the synthetic samples produced by the generator network are transformed back to the space of the original data by the decoder network of the AE.

The reasons for using the hybrid AE+GAN architecture for the DAGGER framework are twofold. First, current efforts for synthetic data generation typically involve either GANs or Variational AEs (VAEs). However, as mentioned, such standalone efforts result in serious practical difficulties. Our hybrid architecture can exploit the strengths of both AEs and GANS, while avoiding their weaknesses: the AE part significantly reduces the dimensionality of the real data, and the GAN produces quality synthetic samples when operating in the low dimensional data space, which helps with its stability issues in the training process. Second, our hybrid architecture directly addresses the main constraints imposed by the application scenario, as the AE part takes care of the high dimensionality of the data, and the GAN part can then operate with a number of defining parameters comparable to that of the number of signals for training (i.e., a few hundreds).

We provide details about the AE and the GAN in the DAGGER framework next.

3.1. Autoencoder (AE)

An AE is an unsupervised learning scheme, in the present typically implemented via neural networks, that are trained to reconstruct their inputs (Bank, Koenigstein, & Giryes, 2020). A standard AE architecture consists of two neural networks, an encoder network and a decoder network. The encoder transforms the input into a lower dimensional signal, "compressing" its information content to a reduced but meaningful set of features, which define what is commonly known as the *latent space* of the AE. The decoder performs the opposite transformation, taking a signal in the latent space and transforming it back out into its original space representation.

For the DAGGER framework the AE network follows a standard architecture, with the encoder formed by an input layer matching the length of the real time series signals (i.e., a few thousands), and a number of hidden linear layers, with rectified linear units (ReLU) as activation functions, that progressively reduce the size of the signal down to a couple of hundreds. The decoder follows exactly the same architecture but in the reverse order.

3.2. Wasserstein Generative Adversarial Network (WGAN)

GANs are semi-supervised and unsupervised learning techniques that attempt to capture the distribution of a set of true examples and generate new (unseen) samples out of it (Creswell et al., 2018; Gui, Sun, Wen, Tao, & Ye, 2021). The most common incarnation of a GAN in the present involves two neural networks arranged adversarially: a generator network that transforms a noise signal into one resembling the true data, and a discriminator network that tries to discriminate between real and synthetic data samples as accurately as possible (Gui et al., 2021). The two networks are trained at the same time and in competition with each other, as the generator tries its best to dupe the discriminator, and the discriminator tries its best not to let that happen (Creswell et al., 2018).

In the DAGGER framework, the GAN part doesn't strictly follow the standard architecture (as first conceived by Goodfellow et al. in 2014 (Goodfellow et al., 2014)), and instead takes the form of a Wasserstein GAN (WGAN) (Arjovsky, Chintala, & Bottou, 2017), which corresponds to a few minor practical modifications (although with deep theoretical connotations) to the traditional GAN operation to alleviate some of its weaknesses (Shmelkov, Schmid, & Alahari, 2018):

- In the traditional GAN the generator tries to minimize $\mathbb{E}_x[\log(D(x))] + \mathbb{E}_z[\log(1 - D(G(z)))]$ (with $D(\cdot)$ representing the discriminator and $G(\cdot)$ the generator, and x and z representing respectively a full data sample and a noise signal), and the discriminator tries to minimize it; in the WGAN, the generator tries to minimize $D(G(z))$, and the discriminator, now called *critic*, tries to minimize $D(x) - D(G(z))$. The reason that the discriminator changes its name in the WGAN is the fact that its output is not longer a number in $[0, 1]$ (with 0.5 as the decision boundary between real or synthetic samples), and instead is any real number, which can be interpreted as a mechanism that rates the "realness" of the samples, instead of simply classifying them between real and synthetic.
- After every gradient update on the critic function (i.e., $D(\cdot)$), the defining weights of the critic are kept bounded

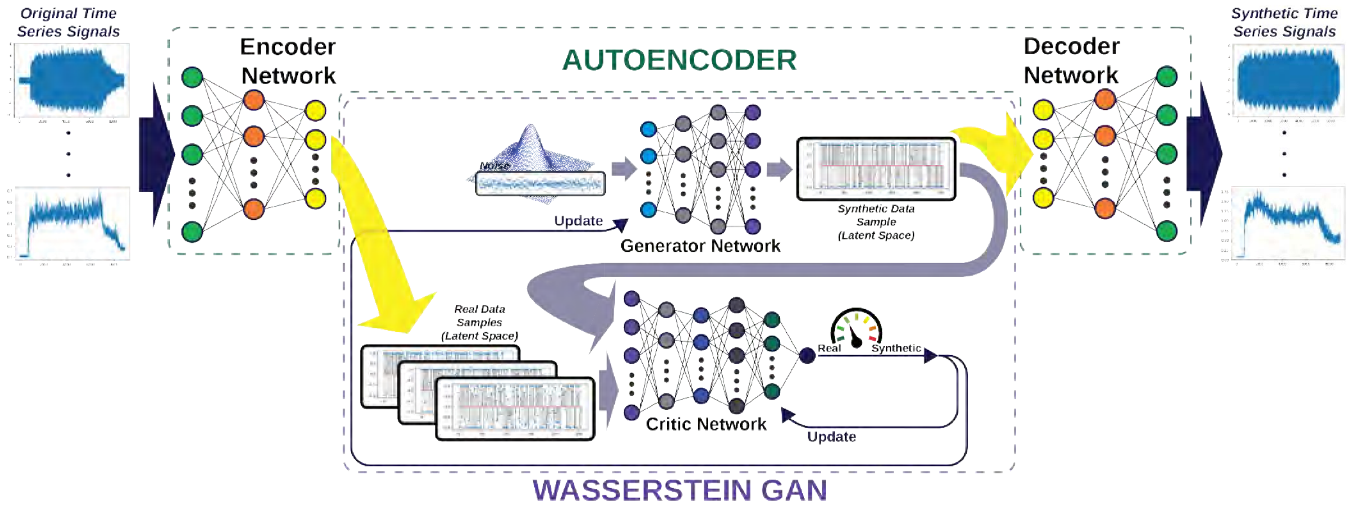

Figure 1. Schematic of DAGGER framework for synthetic time series generation in manufacturing application scenarios.

inside a small range of values, as opposed to allowing them to take any value.

- The optimizer of the critic switches from the traditional momentum based methods, like Adam, to RMSProp, which tries to resolve the problem that gradients may vary widely in magnitudes for the different weights of the critic. It does so by adapting the step size individually for each weight.
- The generator is trained less frequently than the critic is (e.g., 1 training cycle for the generator every 5 training cycles for the critic).

3.3. Training and Querying the DAGGER

The schematic shown in Figure 1 suggests that the two main components of the DAGGER (the AE and the WGAN) operate simultaneously, without much distinction between the specifics of the training process and those of the generation of synthetic samples. Strictly speaking, however, the different parts of the two main networks are employed at different times, depending on whether training or synthetic data generation is taking place, in an asynchronous fashion.

3.3.1. Training

The training of the DAGGER, in the traditional sense of the term (i.e., a process taking place *a priori*, with the sole purpose of finding the "optimal" values for the defining hyperparameters of the model), mainly involves the AE network. In other words, the real time series data is used to train the AE network, involving both the encoder and the decoder, without involving the WGAN. Then, once convergence has occurred (the loss is below an acceptable threshold), the original data is passed through the encoder network of the AE, and its output is then used to "train" the WGAN.

The training of the WGAN, like with most traditional GANs, occurs while synthetic data is being generated, adjusting its hyperparameters at every "training" step. This takes place until an acceptable level of performance is reached. Once that point is reached, the DAGGER can be queried for new synthetic time series samples.

3.3.2. Querying

Querying the DAGGER for synthetic data samples corresponds to simply having the generator of the WGAN produce synthetic latent space samples, and subsequently having the decoder network of the AE transform them from the latent space to the space of the time series data.

4. PERFORMANCE EVALUATION

In this section we present the performance evaluation results for the DAGGER framework. For this we consider two different assessment mechanisms, one directly analyzing the "closeness" of the synthetic data to the real data via numerous evaluation tools (like the means per time sample, the variances per time sample, DFT, KL distance, etc.), and the other one indirectly assessing the quality of the synthetic data through its ability to produce approximate system matrices when performing system identification for input-output signal pairs of a dynamical system.

4.1. Comparison with Riemannian Hamiltonian Variational AE (RHVAE)

Variational autoencoders (VAEs), first introduced by (Kingma & Welling, 2013) and (Rezende et al., 2014), are AE systems that can generate synthetic data by perturbing the original data in the latent space through, for example, random noise addition, and then mapping the resulting data back to the original space via the decoder network. In the VAE research world, a few works have emerged directly tackling the data augmentation problem under the constraints of small training data set and very high dimensional data: (Chadebec, Mantoux, & Allassonnière, 2020; Chadebec & Allassonnière, 2021; Chadebec, Thibeau-Sutre, Burgos, & Allassonnière, 2022). Their general approach is based on improvements to two of the key procedures in the VAE operation, one being the structure of the latent space, and the other one being the sampling procedure carried out in the latent space to generate the synthetic data.

Regarding the geometry of the latent space, the latest of works listed above ((Chadebec et al., 2022)) propose what the authors refer to as the Riemannian Hamiltonian Variational AE (RHVAE), and it configures the latent space as a Riemannian manifold with a specific metric $\mathbf{G}$. Noting that defining the metric using standard avenues (i.e., involving the Jacobian of the generator function) results in considerable computational difficulties in practice, Ref. (Chadebec et al., 2022) defines a scheme to learn the metric $\mathbf{G}$ as a function of an artificial variable z, from the data and the structure of the neural network. To then navigate and sample the resulting Riemannian latent space, (Chadebec et al., 2022) defines a multivariate zero-mean Gaussian random vector v, with covariance matrix given by $\mathbf{G}(z)$. Then, the "exploration" of the Riemannian latent space is based on Hamiltonian dynamics, with z being the position and v the velocity: the Hamiltonian is found by adding the resulting potential energy and kinetic energy, which are functions of z and v. With this, a leapfrog integrator scheme is defined to iteratively compute the evolution in time of z and v from the Hamiltonian expression, which ensures that the target distribution is preserved, and that the update procedure is volume preserving and time reversible. This iterative process then configures a Markov Chain (MC) on z, with transition probabilities given by particular functions of the metric $\mathbf{G}$ (which is in turn a function of the current values of z and v). The defined MC is said to converge to a distribution p_{target}, which is used to efficiently estimate the relevant distributions from where the synthetic data samples are ultimately drawn. This approach is then shown in (Chadebec et al., 2022) via extensive numerical analyses to be able to handle very high dimensional data (since the operation essentially occurs in the latent space, which is considerably lower dimensional), and to effectively operate with very few training samples (as the sampling scheme is defined over more meaningfully constructed distributions).

Considering how well the RHVAE framework fits the problem outlined in Section 2, we utilize it as the main contrast medium for the DAGGER in the performance evaluation subsections below.

4.2. Direct Performance Assessment

To validate the performance of the DAGGER with the RHVAE, six time-series sensors from a milling environment were compiled from (Teubert, 2022) (referenced as mill data set). Each sensor had a training set composed of 146 runs of 9000 length time steps. The majority of the discussion focuses on three signals: AC motor current, spindle vibration, and table vibration. The AC motor current was chosen due to it being the only signal having a periodic nature while the spindle vibration was chosen due to its more random transient properties. The table vibration had similar properties as the remaining three signals with well defined transients and moderate noise being present. A toy data composed of a sinusoid containing a random phase shift and amplitude enveloping per run (referenced as modulated sinusoid) was also used for model training.

The metrics for performance evaluation of the generative models included: mean and variance comparisons, K-L divergence, auto-correlation of mean and variance signal, and discrete Fourier transformations (DFT). These metrics were chosen to better quantify the DAGGER's ability to capture important time series characteristics such as: multiple transient properties, probability density evolution over time, frequency information, and statistical properties. Error between the synthetic and real data was calculated using mean absolute percentage error (MAPE) and is provided in the figures for the auto-correlation section and expected value/variance section of the results.

4.2.1. Statistical Moments

The first statistical analysis technique determines the mean and variance of each signal. This metric allows for quantification of the DAGGER's ability to capture basic statistical properties of the real data sets. Both the variance and expected value were calculated for each time step of the entire data set. This methodology analyzes how well DAGGER captures both transient properties and the evolution of statistical properties as the run progresses.

DAGGER was robust in accurately producing data that captured the spread of possible values for the modulated sinusoids, shown in Figure 2. This is evident from the variance signal generated by DAGGER having 19.8% less MAPE error than the RHVAE. Both models were comparable in capturing the expected value with DAGGER having 3.4% error (0.2% less than RHVAE). Although having almost identical performances, DAGGER produced smoother signals and better captured significant trends present in the real data set's expected value signal.

DAGGER had similar performance to the RHVAE for generating synthetic data when training on the mill data set. Figure 3 displays this with only slight error differences being

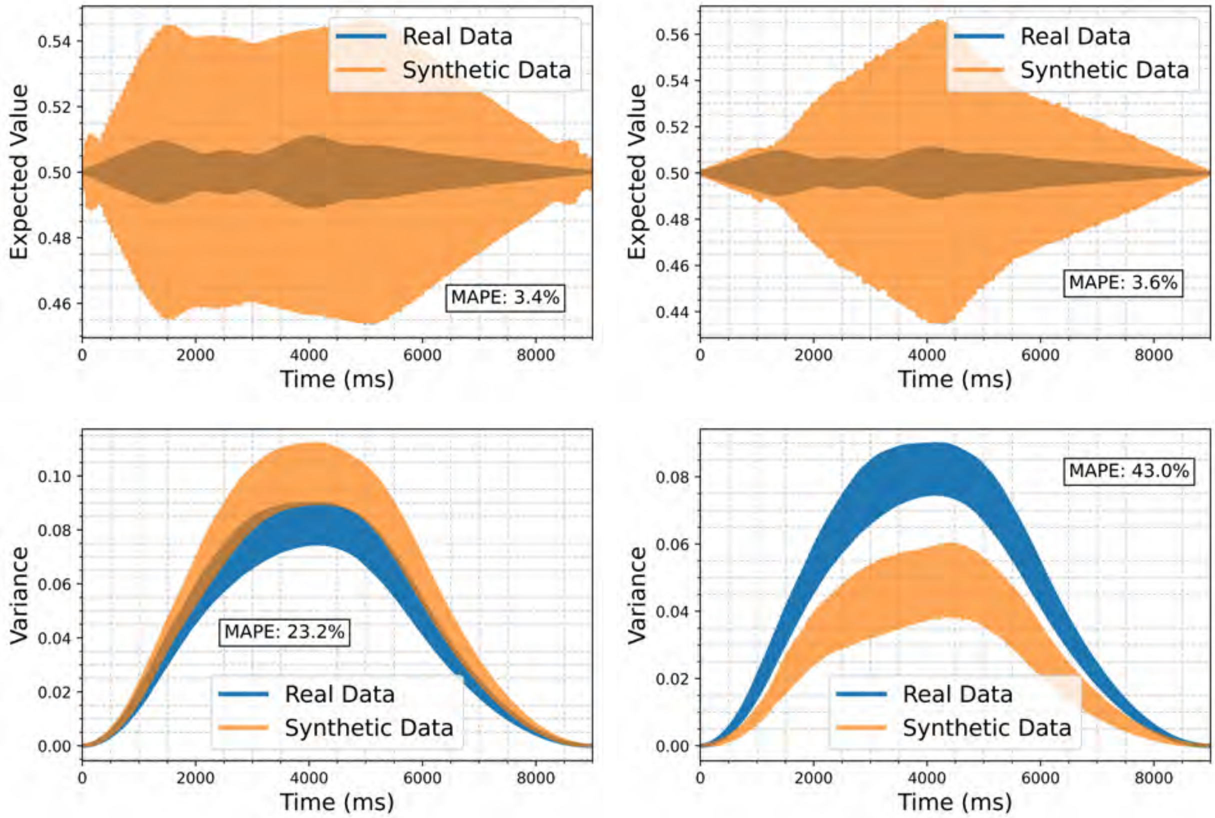

Figure 2. The DAGGER (left column) and RHVAE (right column) generated/real expected value and variance signal for the modulated sinusoid. Note: The graph for the real data is sometimes presented in gray due to the overlap from the graph of the synthetic data.

present for most of the signals. The main exception is with spindle vibration signal with the RHVAE poorly capturing the expected value signal having 27.1% higher error than DAGGER. Excluding that signal, both models performed equally well in capturing major trends present in remaining signals in the mill data set.

In Figure 4 the variance signal of the same three signal shown in Figure 3 are given, DAGGER performed well with capturing trends and variance values for the majority of the signals. DAGGER did have some issues when attempting to capture the variance of the AC motor current as compared to the RHVAE. However, the RHVAE saw issues when modeling the properties of the table vibration and was outperformed by DAGGER. The difference in the nature of these two signals is likely the cause for the noted discrepancies as the AC motor current is a periodic signal while the spindle vibration signal is notably more random and noisy in nature.

4.2.2. KL Divergence

The second metric employed was the Kullback-Leibler (KL) divergence. This metric measures the quality of the synthetic data by comparing how similar the probability densities are for every time step within the training data. This metric allows for the direct comparison of probability distributions over the time evolution of the signal. A KL divergence of zero indicates that the probabilities are identical where as a higher values would indicate less similarity. The bin size for the histograms remained constant for each case and all data sets were normalized ($[0, 1]$) prior to histogram generation.

DAGGER performance was comparable to that of the RHVAE for the modulated sinusoid case with similar trends and values for the KL divergence (Figure 5). Figure 5 depicts similar trends with the main differences being noticed at $t_0 = 0\ ms$ and $t_f = 9000\ ms$. However, DAGGER's performance is slightly improved with the majority of KL divergence values at or below 0.2 while the RHVAE was close to 0.25 for nearly $1000\ ms$.

For four of the six signals, similar performance between the RHVAE and DAGGER was observed. The largest variations occurred with the AC motor current and table vibration as shown by Figure 6. The KL divergence for the AC small motor current was at or below 0.2 for the majority of the synthetic DAGGER data set while the RHVAE kept KL divergence under 0.05 a part from a few time steps. Referencing

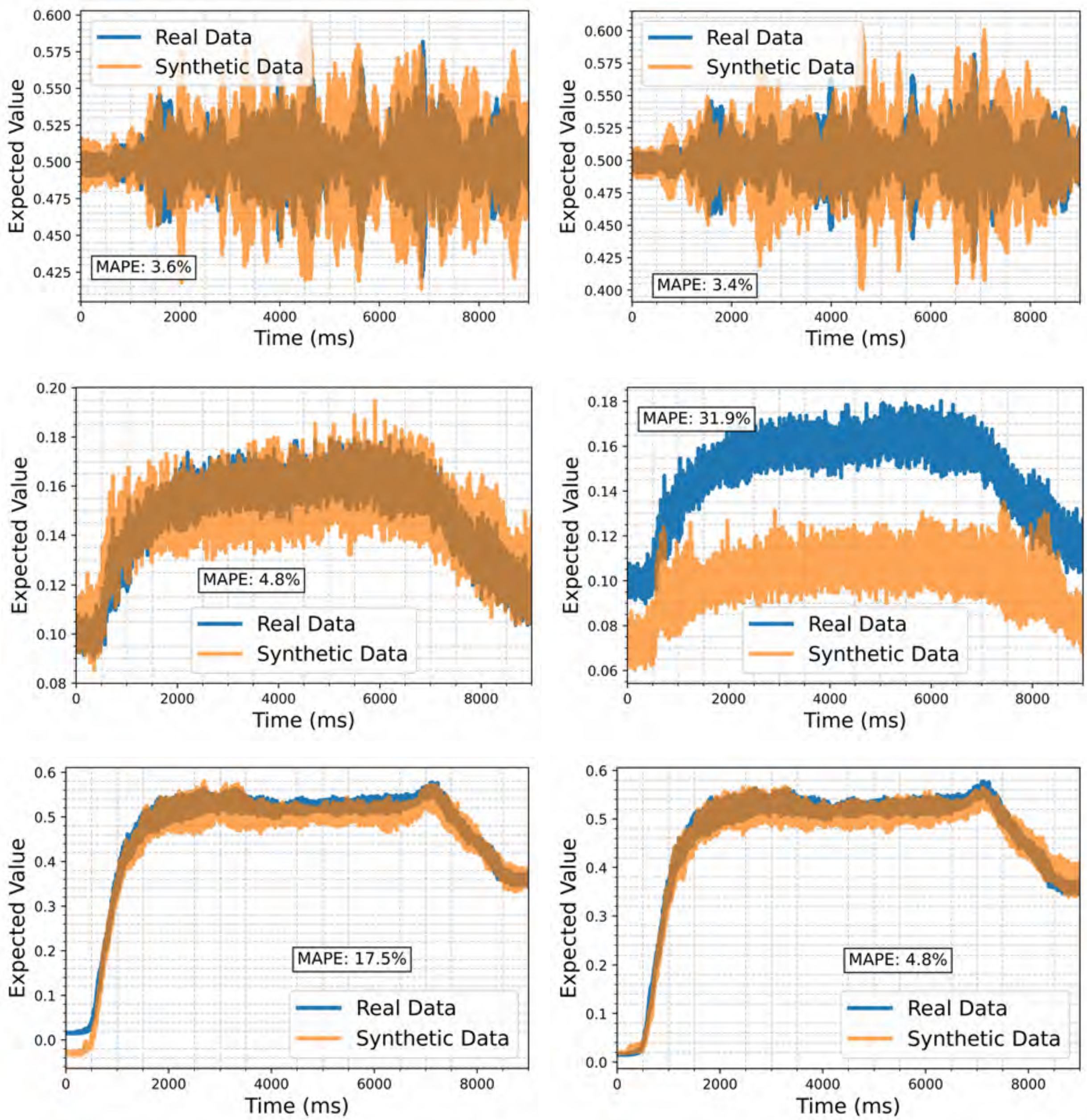

Figure 3. The DAGGER (left column) and RHVAE (right column) generated/real expected value signal for the AC current signal (top row), spindle vibration (middle row), and table vibration (bottom row), present in the mill data set. Note: The graph for the real data is sometimes presented in gray due to the overlap from the graph of the synthetic data.

Figure 4, DAGGER saw issues with capturing the variance of this signal thereby leading to this discrepancy. DAGGER outperformed the RHVAE for the table vibration generations with the KL divergence of DAGGER hovering near 0.1 while the RHVAE was near 0.2.

4.2.3. Auto-correlation

The next metric used for performance measurement was the auto-correlation of the variance and expected value signal. For time series signal cases, auto-correlation shows trends present in the signal as the lag increases. Comparing the auto-correlation of the generated data's expected value and variance signal with that of the real data reveals how well the generative models will be able to capture the change of dynamics over time. The mean absolute percentage error (MAPE) can then be calculated for these signals as a metric for performance between DAGGER and RHVAE via:

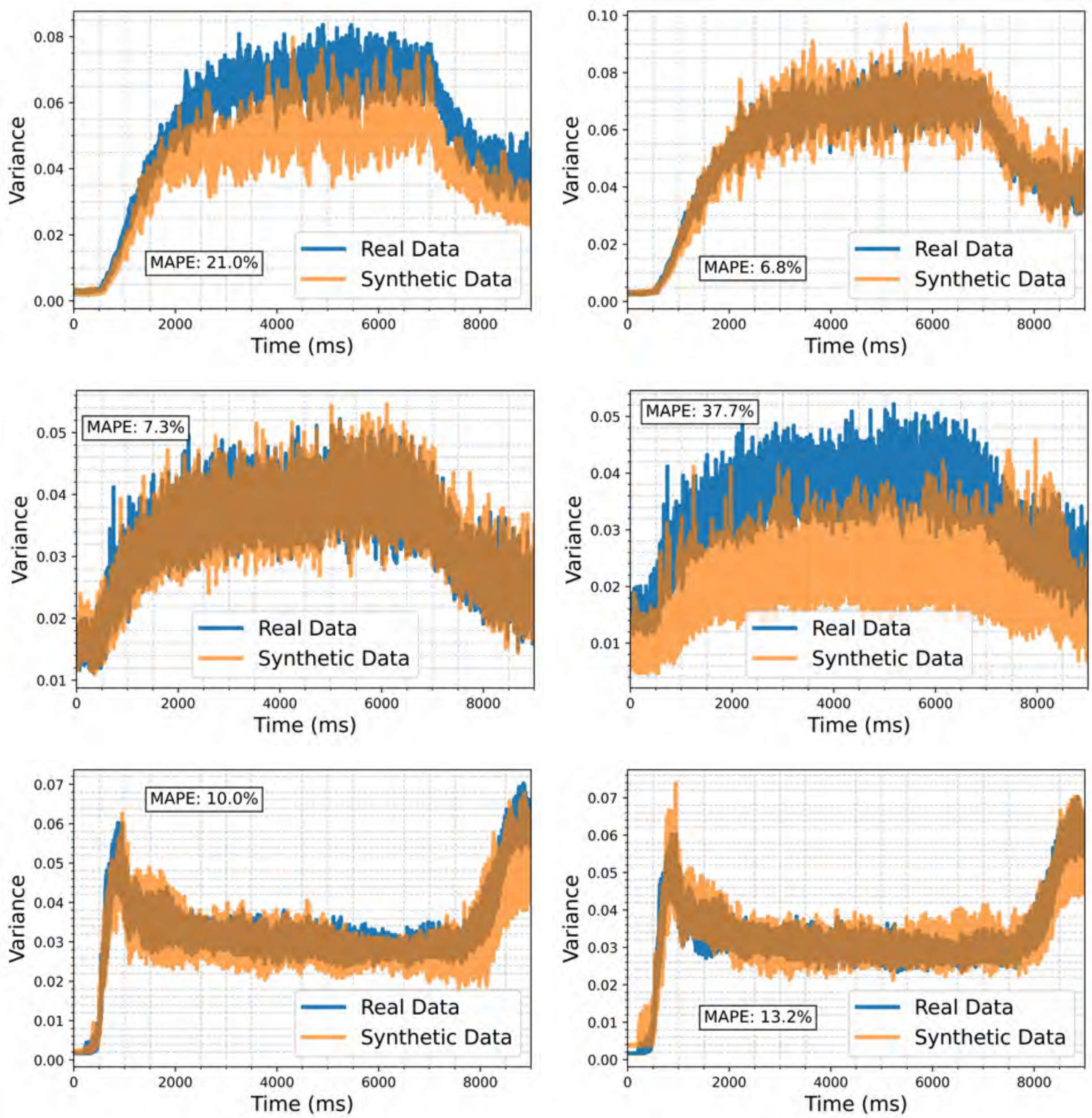

Figure 4. DAGGER (left column) and RHVAE (right column) generated/real variance signal for: AC motor current, spindle vibration, and table vibration, present in the mill data set. Note: The graph for the real data is sometimes presented in gray due to the overlap from the graph of the synthetic data.

$$M = \frac{1}{n} \sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right| \times 100 \qquad (2)$$

where n is the number of simulated time steps, A_t the real signal, and F_t the synthetic data.

Both DAGGER and RHVAE showed issues with capturing the time trend of the modulated sinusoid with MAPE errors above 1000% for the expected value signal (Figure 7). The large MAPE error is caused by the real data having near 0 values for a considerable number of time samples, which translates into near 0 denominator values in the percentage calculations. Both DAGGER and RHVAE show improved performance when capturing trends in the variance, with DAGGER outperforming the RHVAE by having 9.1% less MAPE error. This improvement can be explained through DAGGER having less variance present throughout the auto-correlation signal.

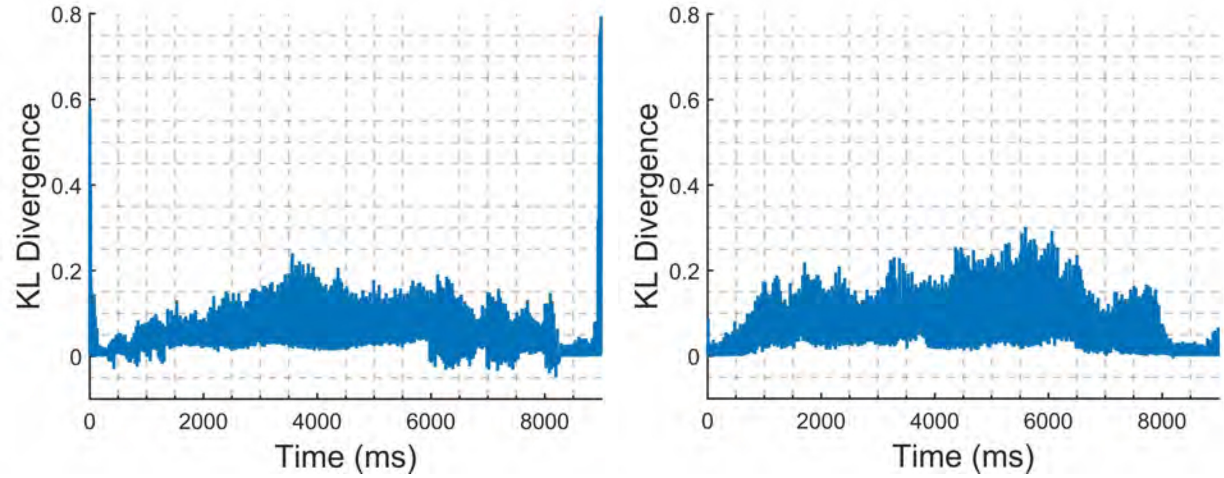

Figure 5. DAGGER (left column) and RHVAE (right column) KL divergence for the modulated sinusoid

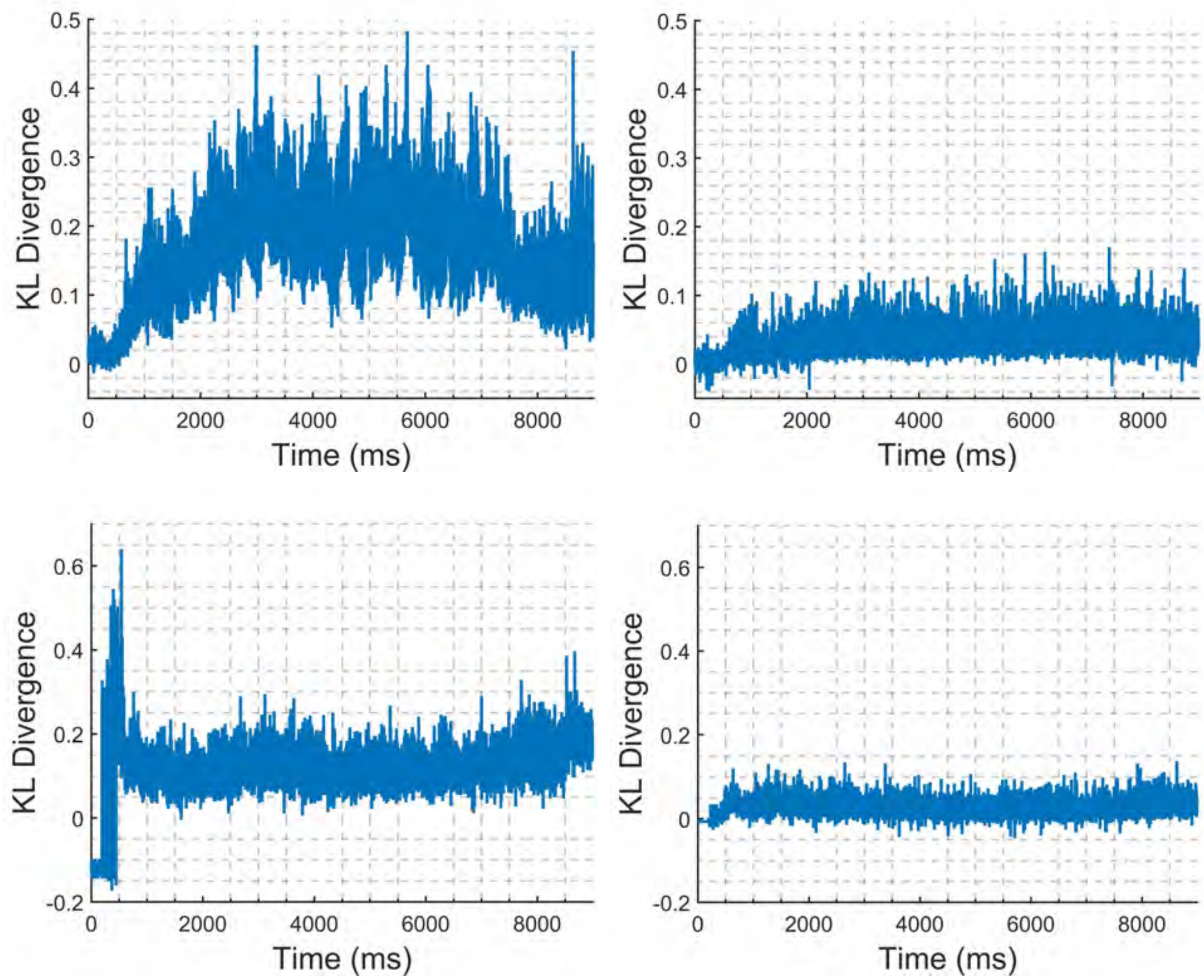

Figure 6. DAGGER (left column) and RHVAE (right column) KL divergence for the AC small motor current (top row) and table vibration (bottom row)

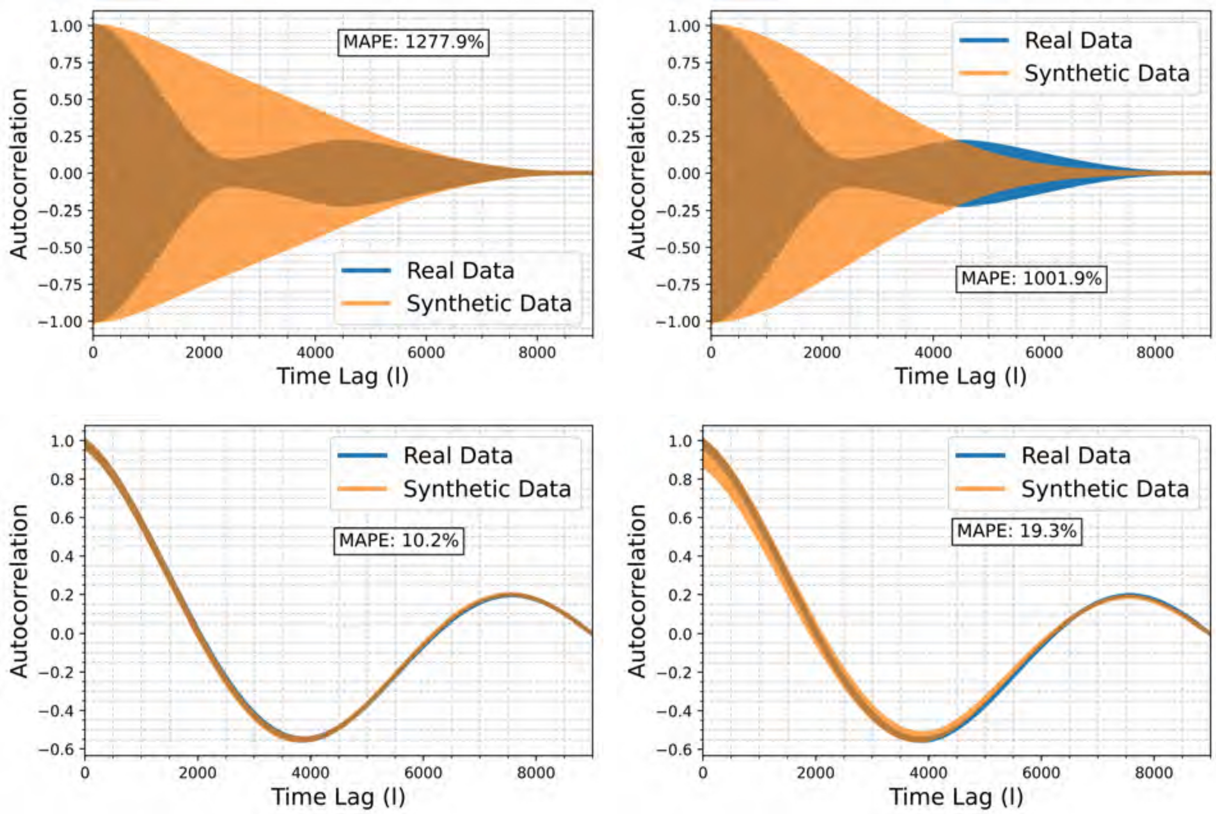

Figure 7. DAGGER (left column) and RHVAE (right column) auto-correlation for the expected value (top row) and variance (bottom row) signals. Note: The graph for the real data is sometimes presented in gray due to the overlap from the graph of the synthetic data.

DAGGER and RHVAE show similar performance in capturing the trends of the expected value for the mill sensors (Figure 8). When capturing trends in the AC motor current, performance degrades with MAPE errors in excess of 600%. As mentioned in the variance signal analysis, this large error is caused by near 0 values in the real data, which goes in the denominator of the error calculations. For the remaining signals, both models demonstrate nearly identical performance capturing the lagged trends of the expected value signal. DAGGER's and RHVAE's auto-correlated expected value signals reflect noisy trends when compared with the training data set for the spindle vibration. This was most likely caused from high noise content present throughout the spindle vibration signal. The remaining signals had similar auto-correlated expected value signals as the table vibration. In general, DAGGER and RHVAE had similar performance in both capturing signal trends and similar MAPE errors for these signals.

Shown in (Figure 9), DAGGER maintained similar performance to the RHVAE in capturing the lagged trends in the variance for both the AC motor current and table vibration. DAGGER captured the lagged trends of the variance for the noisy spindle vibration signal with 68.7% less MAPE error than the RHVAE.

4.2.4. DFT

The final metric used was the discrete Fourier transform. This was chosen due to the DFT's ability to transform a time series signal into a frequency domain representation that can capture both frequency and amplitude information. Using this metric on the synthetic and real data allows for a visual comparison of the performance of DAGGER in capturing frequency information.

DAGGER captured the main frequency and amplitude information present throughout the training set, but incorrectly excluded the high frequency behavior (Figure 10). This performance was comparable to the RVHAE with both capturing the low frequency behavior. Both DAGGER and RHVAE also contain a low amplitude noise floor that is not present throughout the modulated sinusoid.

One should note DAGGER's ability in capturing the major frequency and amplitude components present in the mill signal as shown by Figure 11. DAGGER and RHVAE performed

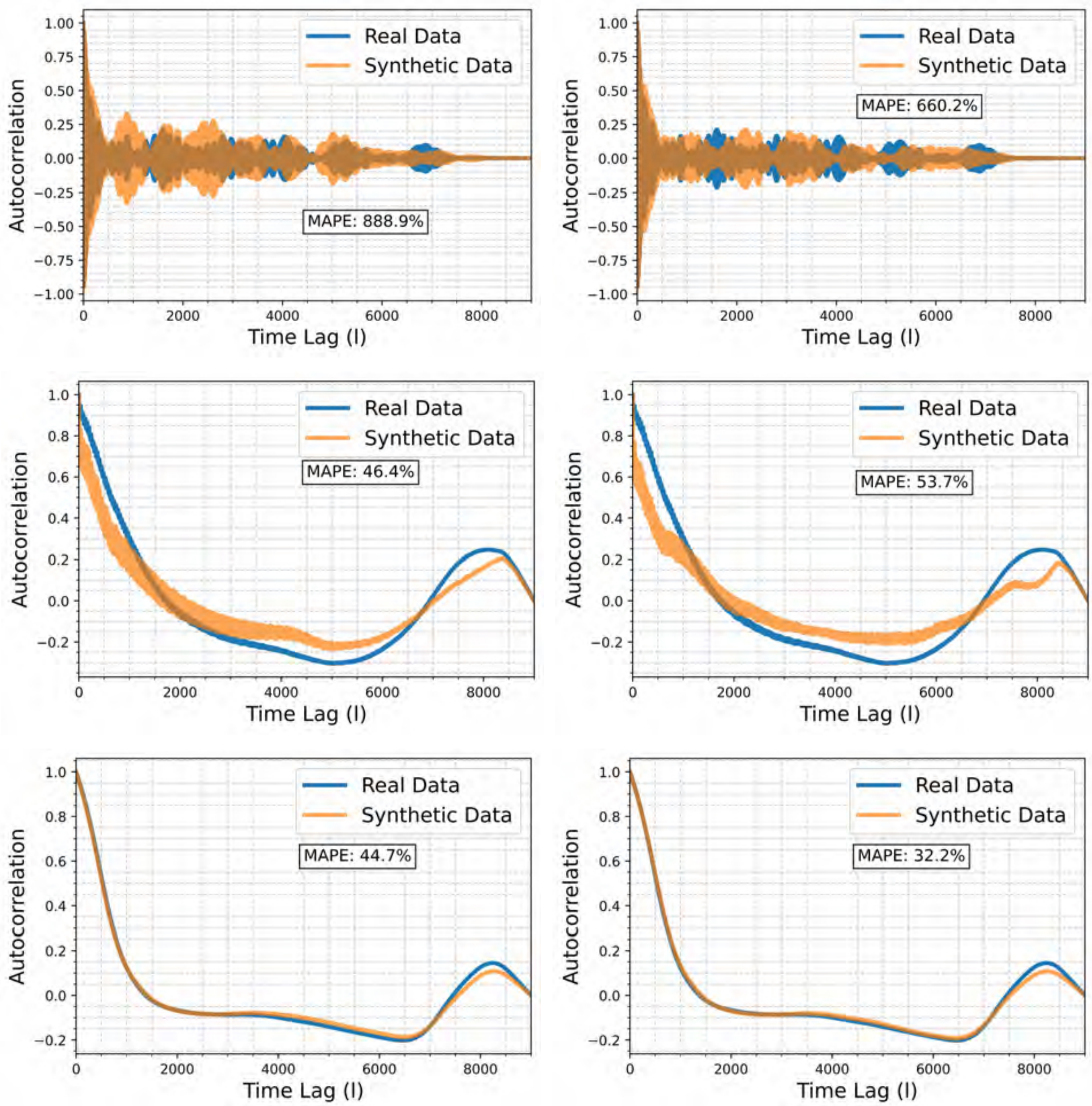

Figure 8. DAGGER (left column) and RHVAE (right column) generated/real auto-correlation signal for the expected value signal of the AC motor current (top row), spindle vibration (middle row), and table vibration (bottom row). Note: The graph for the real data is sometimes presented in gray due to the overlap from the graph of the synthetic data.

similarly for the AC motor current with a DC component (caused by data normalization) and the main AC current frequency being depicted. Both models also have shown similar performance in capturing the high and low frequency components present in the table vibration signal.

4.3. Indirect Performance Assessment

Here we consider the quality of the synthetically generated data by measuring its effectiveness when building black box models out of it. For this we use the model of a simple dynamic system typically studied in the context of control theory, and then use synthetic input-output signal pairs generated by DAGGER and RHVAE to perform system identification. The quality of the synthetic data is established by comparing the resulting model matrices with those built with the original data.

The direction to *take two pills every* 8 *hours* can be considered as a control problem and is modeled via a system

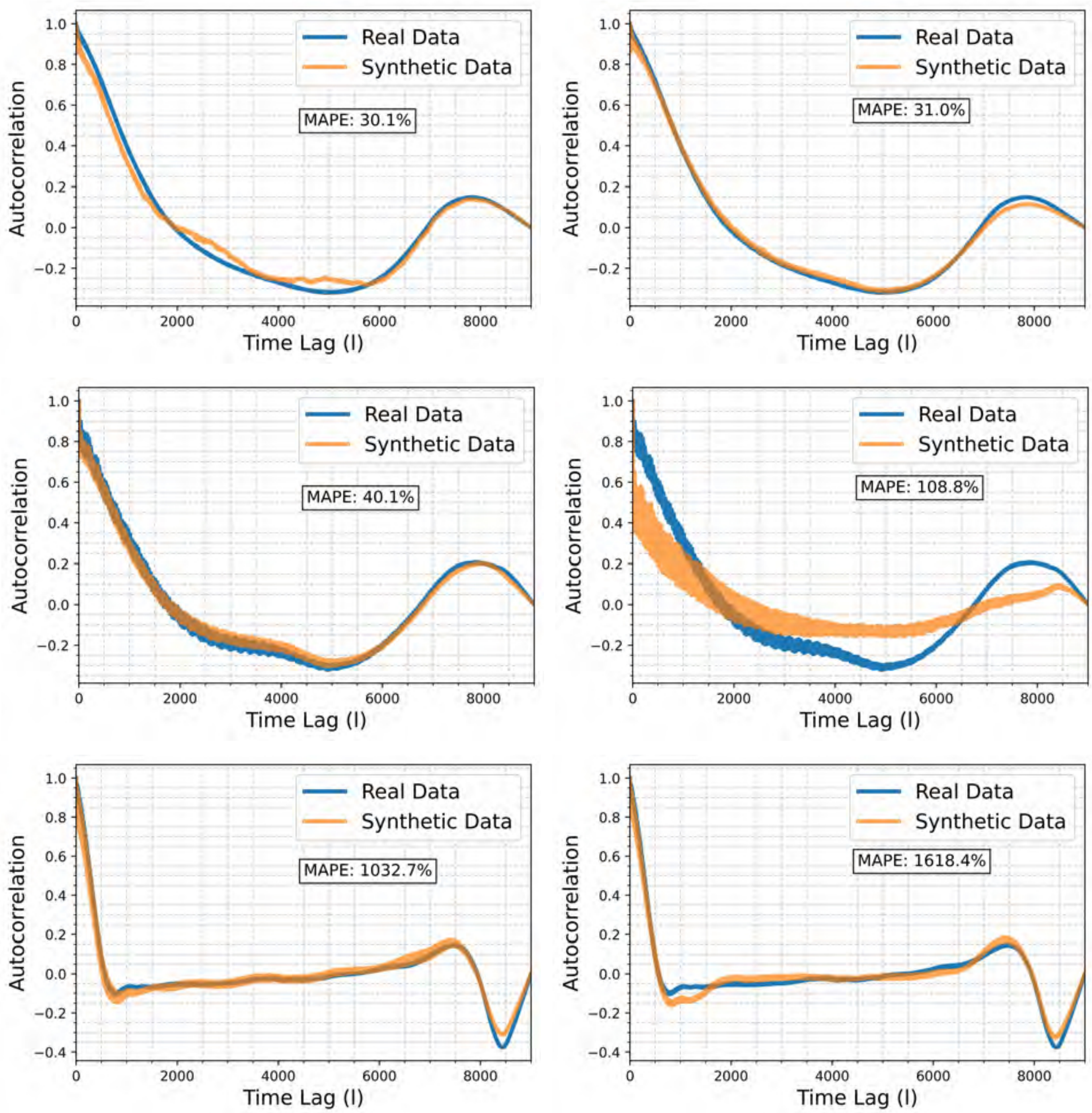

Figure 9. DAGGER (left column) and RHVAE (right column) generated/real auto-correlation signal for the variance of the AC motor current (top row), spindle vibration (middle row), and table vibration. Note: The graph for the real data is sometimes presented in gray due to the overlap from the graph of the synthetic data.

known as compartment models (Åström & Murray, 2010). A schematic to illustrate the idea of the model is shown in Figure 12 (Åström & Murray, 2010). Major components of the human body such as the blood, tissues, and lungs are viewed as compartments separated by membranes. The flow rates between compartments are proportional to the concentration differences in each compartment.

Simplifying the left half of Figure 12 into two compartments as well as assuming there is perfect mixing between compartments and that transport is driven by concentration differences yields (Åström & Murray, 2010):

$$V_1 \frac{dc_1}{dt} = q(c_2 - c_1) - q_0 c_1 + c_0 u, \quad c_1 \geq 0, \tag{3a}$$

$$V_2 \frac{dc_2}{dt} = q(c_1 - c_2), \quad c_2 \geq 0, \tag{3b}$$

$$y = c_2 \tag{3c}$$

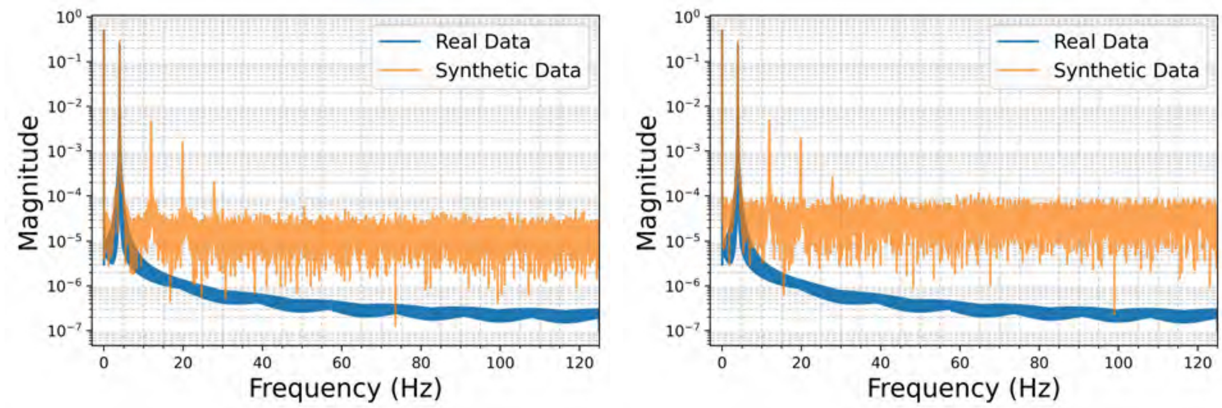

Figure 10. DAGGER (left column) and RHVAE (right column) discrete Fourier transform for the synthetic data of the modulated sinusoid. Note: The graph for the real data is sometimes presented in gray due to the overlap from the graph of the synthetic data.

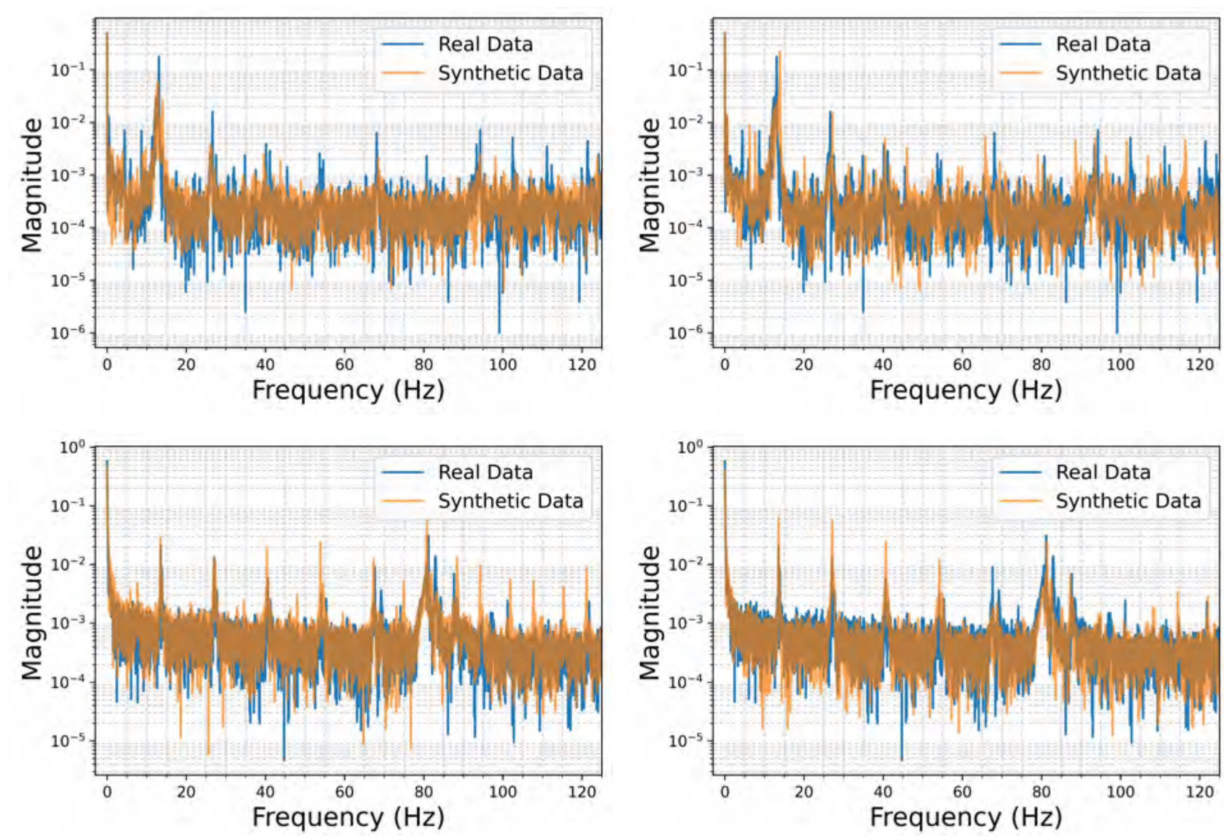

Figure 11. DAGGER (left column) and RHVAE (right column) discrete Fourier transform for the synthetic data for the AC motor current (top row) and table vibration (bottom row). Note: The graph for the real data is sometimes presented in gray due to the overlap from the graph of the synthetic data.

where c_1 and c_2 are drug concentrations in each compartment and V_1 and V_2 are the compartment volumes, which is shown on the right half of Figure 12. Equation (3) can be written in state-space form through the introduction of $k_0 = q_0/V_1$, $k_1 = q/V_1$, $k_2 = q/V_2$, and $b_0 = c_0/V_1$ giving (Åström & Murray, 2010):

$$\frac{dc}{dt} = \begin{bmatrix} -k_0 - k_1 & k_1 \\ k_2 & -k_2 \end{bmatrix} c + \begin{bmatrix} b_0 \\ 0 \end{bmatrix} u, \quad y = \begin{bmatrix} 0 & 1 \end{bmatrix} c \quad (4)$$

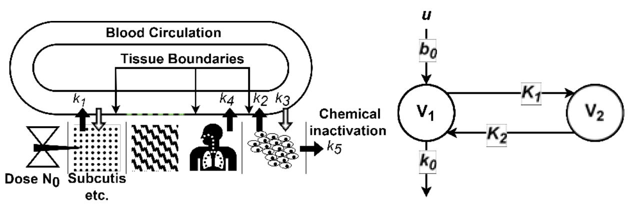

Figure 12. Compartment Modeling Schematics

For the test case here, $k_0 = 0.1$, $k_1 = 0.1$, $k_2 = 0.5$, $b_0 = 1.5$, $c_1 = 0$, and $c_2 = 1$. Four random instances (experiments) of the real normalized $[0, 1]$ input signal u are depicted in top left of Figure 13 whereby a sinusoid of random amplitude between $a : [1, 5]$, random phase $\phi : [0, 1/3\pi]$ and increasing frequency $\omega : [1, 2\pi \pm 1/4\pi]$ for a simulated mixing duration of 50 min is defined. The corresponding output signal of the data is shown on the upper right. Similarly, four random instances of the synthetic normalized input-output signals for the DAGGER and RHVAE platforms are shown on in the middle and bottom row of Figure 13. As both the real and synthetic input signals are random, an indirect quantitative comparison will be performed here. However, looking closely at the DAGGER and RHVAE signals, note the noisy behavior of both the inputs and outputs as compared to the real data set. To perform a indirect comparison, one can turn to system identification to generate corresponding matrix equations of similar form to Eq. (4).

For the system identification problem, 150 experimental input signals were generated using the sinusoidal inputs as defined above and passed through Eq. (4). It should be noted that in order to generate both the input signal and output data simultaneously through DAGGER and RHVAE for system identification, the signals were concatenated. This approach was chosen to maintain some connectivity in the system defined by Eq. (4). This data set was then treated as the seeds for training DAGGER and RHVAE. Subsequently, the platforms generated 150 augmented experimental signals, a few of which are shown in Figure 13. System identification attempts to identify the A, B, C, and D matrices shown by the generic state-space form given below:

$$\dot{x} = Ax + Bu \tag{5a}$$

$$y = Cx + Du \tag{5b}$$

with $A = \begin{bmatrix} -k_0 - k_1 & k_1 \\ k_2 & -k_2 \end{bmatrix}$, $B = \begin{bmatrix} b_0 \\ 0 \end{bmatrix}$, $C = \begin{bmatrix} 0 & 1 \end{bmatrix}$, and $D = 0$ for the real data generated via Eq. (4). Leveraging the DAGGER and RHVAE experimental data and a generic subspace method within the time-domain for system identification, the platforms' A, B, C, and D matrices can be compared to the real matrices also passed through an identical system identification algorithm. It should be noted that both the real and synthetic data sets were normalized between $[0, 1]$, which explains the mismatch between the matrices given by Eq.(4) and those shown here in Table 1.

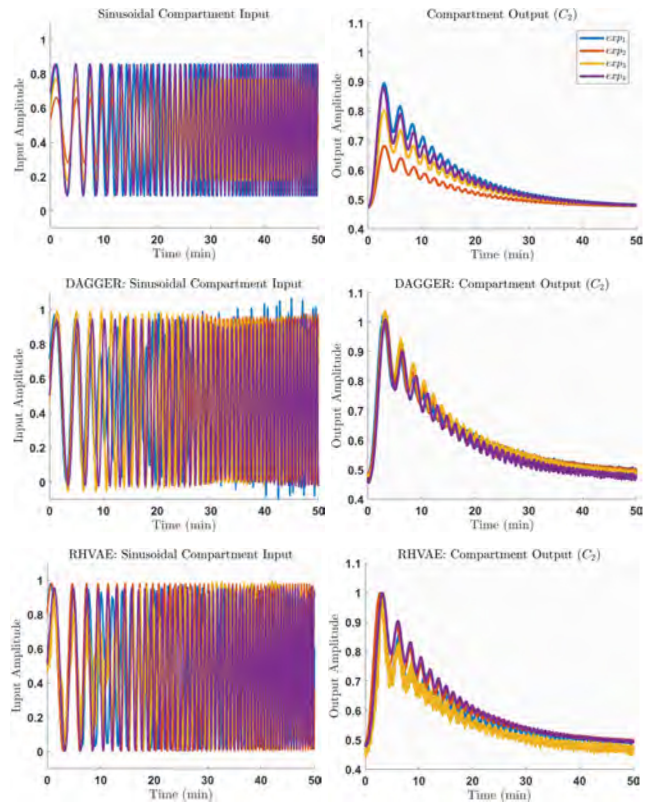

Figure 13. Compartment Model (Eq. (3)) Input-Output Visualization

By inspection, the A, and B matrices appear relatively close to another. In fact, the distance between A_{Real} and A_{DAG} as measured by the Frobenius norm is: $||A_{Real} - A_{DAG}||_{\mathcal{F}} = 0.0719$ and $||B_{Real} - B_{DAG}||_{\mathcal{F}} = 1.8311 \times 10^{-4}$. This corresponds to DAGGER adequately capturing the sinusoidal input signal. Similarly, for the RHVAE $||A_{Real} - A_{RH}||_{\mathcal{F}} = 0.0397$ and $||B_{Real} - B_{RH}||_{\mathcal{F}} = 8.3009 \times 10^{-5}$ where one could also conclude the input signal has been adequately captured. In contrast, $||C_{Real} - C_{DAG}||_{\mathcal{F}} = 420.0466$ and $||C_{Real} - C_{RH}||_{\mathcal{F}} = 422.1737$ meaning the output signal is partially degraded when compared to the result from the input. This is likely due to the required concatenation of the input-output signal prior to training DAGGER and RHVAE causing an abrupt transition between signal behavior at the junction between the left and right hand sides of Figure 13. While the networks can be trained independently for both the input and output signals, one would lose the corresponding connectivity required for system identification. That is, independent networks would match random inputs with random outputs thereby violating the dependence needed for system identification.

Table 1. State-Space Comparison: Real vs. DAGGER vs. RHVAE

	Real	**DAGGER**	**RHVAE**
A	$\begin{bmatrix} 0.9956 & 0.004 \\ -0.011 & 1.0014 \end{bmatrix}$	$\begin{bmatrix} 0.9996 & 0.0113 \\ -0.0044 & 0.9303 \end{bmatrix}$	$\begin{bmatrix} 0.9996 & -0.0044 \\ 0.0119 & 0.9703 \end{bmatrix}$
B	$\begin{bmatrix} -5.4097 \times 10^{-5} \\ -1.2996 \times 10^{-4} \end{bmatrix}$	$\begin{bmatrix} -2.4676 \times 10^{-6} \\ 4.5728 \times 10^{-5} \end{bmatrix}$	$\begin{bmatrix} -5.3345 \times 10^{-7} \\ -6.6542 \times 10^{-5} \end{bmatrix}$
C	$[-112.1032 \quad -45.5782]$	$[305.0968 \quad -3.2411]$	$[307.7099 \quad 0.9949]$
D	$[0]$	$[0]$	$[0]$

5. CONCLUSION

In this paper, we presented the DAGGER framework, which corresponds to a hybrid AE+GAN deep neural network architecture specifically conceived to produce synthetic high dimensional time series data with very small training data sets, which are characteristic of traditional manufacturing applications.

We evaluated the effectiveness of the DAGGER framework by analyzing the quality of the synthetically generated data through both direct and indirect performance assessment methods. In the former, for toy artificial and real sensor data (a publicly available benchmarking data set containing sensor readings from a milling machine), we employed performance metrics based on the similarity between original data and synthetic data in terms of mean, variance, KL divergence, behavior in Fourier domain, and auto-correlation. Regarding the indirect performance assessment, we evaluated the adequacy of the synthetically generated time series samples to perform system identification of a black box dynamic system. In all the cases, we compared the results from DAGGER with those from a RHVAE, which is a recently published data augmentation framework specifically designed to handle very small, high dimensional data sets.

Direct analysis methods revealed similar performance between the DAGGER and RHVAE regarding ability to capture frequency, statistical, and time trend information. Both models captured the transient properties present in the signal with low MAPE error but had difficulty capturing trends in the expected value and variance for noisy signals. Overall, the DAGGER had no significant performance differences in comparison to the RHVAE.

Regarding the indirect performance assessment procedures, DAGGER and RHVAE demonstrated similar results when identifying a compartment model system. Both platforms adequately captured input signal behavior at the detriment to output signal performance. Aside from further hyperparameter tuning within both DAGGER and RHVAE, an alternative to the abrupt transition between the concatenated input-output signals required for system identification will also be addressed to improve synthetic data generation. The similarly between DAGGER and the RHVAE results is supported throughout the direct performance assessment as well whereby both platforms exhibit comparable characteristics during the examination of different synthetically generated signals. That is, the results showed that DAGGER's performance is in general satisfactory, and comparable with that of the RHVAE in all the considered evaluation scenarios. From our tests, the area DAGGER performed better was in the duration of the training process, with it achieving convergence empirically in at least an order of magnitude less time then the RHVAE. This requires a deep and careful exploration, however, and it is thus part of our future explorations.

Looking ahead, optimization of the DAGGER architecture is planned with implementing lower reconstruction error autoencoder networks and tuning parameters of the GAN. Furthermore, software tools will be completed to advance improvements in analyzing techniques to increase understanding of latent space modeling and exploration for the GAN. Different generative models beside the RHVAE will be explored as a benchmark against improved DAGGER architecture.

ACKNOWLEDGMENT

This work is supported by the US Air Force AFWERX SBIR Program under Grant No. FA8649-22-P-0701.

REFERENCES

Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning* (pp. 214–223).

Åström, K., & Murray, R. (2010). *Feedback systems: An introduction for scientists and engineers*. Princeton University Press.

Bank, D., Koenigstein, N., & Giryes, R. (2020). Autoencoders. *arXiv preprint arXiv:2003.05991*.

Chadebec, C., & Allassonnière, S. (2021). Data augmentation with variational autoencoders and manifold sampling. In *Deep generative models, and data augmentation, labelling, and imperfections* (pp. 184–192). Springer.

Chadebec, C., Mantoux, C., & Allassonnière, S. (2020). Geometry-aware hamiltonian variational auto-encoder. *arXiv preprint arXiv:2010.11518*, 0-44.

Chadebec, C., Thibeau-Sutre, E., Burgos, N., & Allassonnière, S. (2022). Data augmentation in high dimensional low sample size setting using a geometry-based variational autoencoder. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., & Bharath, A. A. (2018). Generative adversarial networks: An overview. *IEEE signal processing magazine*, *35*(1), 53–65.

Demir, S., Mincev, K., Kok, K., & Paterakis, N. G. (2021). Data augmentation for time series regression: Applying transformations, autoencoders and adversarial networks to electricity price forecasting. *Applied Energy*, *304*, 117695.

Diez-Olivan, A., Del Ser, J., Galar, D., & Sierra, B. (2019). Data fusion and machine learning for industrial prognosis: Trends and perspectives towards industry 4.0. *Information Fusion*, *50*, 92–111.

Doersch, C. (2021). *Tutorial on variational autoencoders.*

Figueira, A., & Vaz, B. (2022). Survey on synthetic data generation, evaluation methods and gans. *Mathematics*, *10*(15). doi: 10.3390/math10152733

Goodfellow, I. (2016). Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, & K. Weinberger (Eds.), *Advances in neural information processing systems* (Vol. 27). Curran Associates, Inc.

Gui, J., Sun, Z., Wen, Y., Tao, D., & Ye, J. (2021). A review on generative adversarial networks: Algorithms, theory, and applications. *IEEE transactions on knowledge and data engineering*.

Hutter, F., Lücke, J., & Schmidt-Thieme, L. (2015). Beyond manual tuning of hyperparameters. *KI-Künstliche Intelligenz*, *29*(4), 329–337.

Iglesias, G., Talavera, E., González-Prieto, Á., Mozo, A., & Gómez-Canaval, S. (2023). Data augmentation techniques in time series domain: a survey and taxonomy. *Neural Computing and Applications*, *35*(14), 10123–10145.

Khanuja, H. K., & Agarkar, A. A. (2023). Towards gan challenges and its optimal solutions. *Generative Adversarial Networks and Deep Learning: Theory and Applications*.

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. *arXiv*.

Rezende, D.J., Mohamed, S., Wierstra, & D. (2014). Stochastic backpropagation and approximate inference in deep generative models. *International conference on machine learning*, 1278–1286.

Shao, H., Yao, S., Sun, D., Zhang, A., Liu, S., Liu, D., ... Abdelzaher, T. (2020). Controlvae: Controllable variational autoencoder. In *International conference on machine learning* (pp. 8655–8664).

Shmelkov, K., Schmid, C., & Alahari, K. (2018). How good is my gan? In *Proceedings of the european conference on computer vision (eccv)* (pp. 213–229).

Smith, K. E., & Smith, A. O. (2020). *Conditional gan for timeseries generation.*

Teubert, C. (2022). *Milling wear data set.* Retrieved from `https://data.nasa.gov/Raw-Data/Milling-Wear/vjv9-9f3x` (Dataset)

Wright, L., & Davidson, S. (2020). How to tell the difference between a model and a digital twin. *Advanced Modeling and Simulation in Engineering Sciences*, *7*(1), 1–13.

Yang, Z., Li, Y., & Zhou, G. (2023). Ts-gan: Time-series gan for sensor-based health data augmentation. *ACM Transactions on Computing for Healthcare*, *4*(2), 1–21.