NATIONAL RESEARCH UNIVERSITY
HIGHER SCHOOL OF ECONOMICS
International College of Economics and Finance

Timur Galiaskarov

Imputing Missing Values in Financial Time Series Using Generative Adversarial Networks

Term paper
38.04.01 ECONOMICS
Master's Programme **"Financial Economics"**

Moscow 2025

Contents

1 Introduction

Missing data is a prevalent issue in financial time series analysis, often arising from irregular trading hours, system errors, or asynchronous data reporting across markets. These gaps can significantly impair the accuracy of downstream tasks such as forecasting, volatility estimation, and algorithmic trading. Many of these gaps need to be filled. In statistics, this process of replacing missing values is called imputation. Traditional imputation methods, such as linear interpolation, forward fill, and statistical models like ARIMA, often fail to capture the intricate temporal dependencies and stochastic nature of financial data, especially under conditions of high volatility and non-stationarity.

In recent years, deep generative models, particularly Generative Adversarial Networks (GANs), have emerged as powerful tools for learning complex data distributions. Originally proposed for image synthesis, GANs have since been adapted to sequential and time-dependent data, offering a promising framework for imputing missing values in financial time series. Unlike conventional approaches, GANs can learn the underlying joint distribution of observed and missing values, producing more realistic, context-aware and fastest imputations.

This paper explores the application of GAN-based architectures for missing-value imputation in financial time series. We investigate how these models compare with traditional and modern deep learning techniques in terms of imputation accuracy and their ability to preserve the temporal and statistical characteristics of financial data. Our goal is to demonstrate that GANs can offer a robust, data-driven alternative to deal with incomplete financial data sets - an increasingly critical challenge in the era of high-frequency and algorithmic trading. This paper observed research question as how effective is TimeGAN in imputing missing values in financial time series data compared to traditional statistical and machine learning-based imputation methods? Metric which will be use Root mean square deviation(RMSE).

2 Background

2.1 Missing values

In statistical analysis, particularly in time series modeling, the mechanism by which data become missing plays a crucial role in determining the appropriate method to handle such gaps. Financial time series data are especially prone to missing values due to irregular market activity, asynchronous trading between time zones, data reporting delays, or technical failures. Understanding the mechanism behind missingness is essential, as it directly affects the validity of imputation strategies and subsequent inferences.

Rubin's classification (1976) provides a widely accepted framework, dividing missing data mechanisms into three categories: Missing Completely at Random - MCAR, Missing at Random (MAR), and Missing Not at Random - MNAR. Under the MCAR assumption, the probability that a value is missing is independent of both observed and unobserved data. That is, the missingness occurs entirely at random and does not contain any informative content about the data structure. The MAR mechanism occurs when the probability of a value being missing is conditionally dependent on the observed data but independent of the actual unobserved (missing) values. Data are classified as MNAR when the probability of missingness is directly related to the unobserved values themselves, even after accounting for all observed data.

2.2 Traditional Imputation Techniques

2.2.1 Constant imputation

Statistics had many ways to solve the imputation problem without machine learning applications. In this section we will observe most significant of them. First thing we can do is remove the missing value. This method is the easiest way to handle missing data but it is not generally advised as it can lead to loss of useful information and potentially biased results if the missing data is not MCAR. In the result data set became shorter, deletion can introduce bias and reduce statistical power. More sophisticated method to replace Nans to constant such as zero or statistical parameters(Mean/Median/Mode). While simple, this method risks distorting the underlying data distribution and can lead to biased parameter estimates.

Constant imputation is generally discouraged unless there is a justified reason for selecting a specific constant. Otherwise, Mean/Median/Mode imputation is one of the most frequently used methods to handle missing data. In this method, the missing values are replaced with the mean, median, or mode of the variable. Mean imputation is used for continuous variables, while median and mode imputation is used for ordinal and categorical variables, respectively. The problem is the same - they all ignore temporal or structural relationships; underestimate variance; inappropriate for data with trends or seasonality. Sometimes missing values could be filled by last or future observation. Last Observation Carried Forward(LOCF) and Next Observation Carried Backward (NOCB) These methods propagate adjacent observed values forward or backward in time to fill missing values. LOCF uses the most recent past value, while NOCB uses the next available future value. These approaches are particularly common in financial time series where the last known value is assumed to persist temporarily.

2.2.2 Moving average

Alternative common way is using moving average(MA). Rolling Statistics Imputation is a non-parametric, window-based method that estimates missing values by applying a summary statistic, such as the mean, median, or weighted average. In time series contexts, this approach is often motivated by the assumption that temporally adjacent values are more likely to be similar. Financial data such as asset prices, returns, or volumes often exhibit non-stationarity, volatility clustering, and long memory properties. Rolling statistics are most appropriate for short gaps in moderately stable segments of the series, especially when quick preprocessing or low-latency computation is required (e.g., real-time trading systems). However, they are not suitable for structural breaks, regime shifts, or long sequences of missing data so it be useful for a quick data imputation. Let $\{X_t\}$ be a univariate time series with missing values at index t. A rolling window of size w (typically symmetric around t or trailing) is used to compute a local statistic based only on observed values. We choose a rolling mean:

$$\hat{X}_t = \frac{1}{|W_t|} \sum_{i \in W_t} \hat{X}_i, \quad where W_t = \{i : |i - t| \leq w, X_i observed\}$$

This estimate replaces the missing value at t

2.2.3 Interpolation

Interpolation is a class of deterministic methods used to estimate missing values by leveraging the known data points surrounding them. In time series analysis, interpolation is particularly valuable because of the inherent temporal ordering of data. The key assumption underlying these methods is that the underlying process evolves smoothly over time, such that missing values can be reasonably inferred from adjacent observations. General in time series applying three types interpolation liner, spline and polynomial.

We will use linear interpolation, which assumes that the data varies linearly between two known values. Given two observed data points at times t_1 and t_2 , with missing values in between, the missing value at time $t \in (t_1, t_2)$ is estimated as:

$$\hat{X}_t = X_{t1} + \frac{t - t_1}{t_2 - t_1}(X_{t_2} - X_{t_1})$$

2.2.4 The Kalman filter

If $F_t = \{y_t | t = 1, ..., T\}$ - the information available at time t(inclusive) and assume that model is known, including all parameters. Filtering means to recover the state variable μ_t given F_t, that is, to remove the measurement errors from the data.[1] The Kalman filter is a recursive algorithm used for optimal state estimation in dynamic systems. It is particularly effective for imputing missing values in noisy and time-dependent data, such as financial time series, by leveraging both model-based prediction and observed data. The Kalman filter assumes that the underlying data-generating process can be described as a state-space model:

State Equation (system dynamics): $x_t = Ax_{t-1} + w_t, w_t \sim N(0, Q)$

Observation Equation (measurement model): $y_t = Hx_t + v_t, v_t \sim N(0, R)$

Here,x_t represents the latent state (e.g., true price level), y_t is the observed data (e.g., closing price), A is the transition matrix, and H is the observation matrix. The noise terms w_t and vt represent process and measurement noise, respectively. When values in the observation vector yt are missing, the Kalman filter can still proceed by propagating estimates based

[1]RUEY S. TSAY Analysis of Financial Time Series

on the system dynamics, filling in gaps using the prior distribution over the latent state x_t. In general Kalman equation looks like this

$$x_{k+1}^{opt} = Kz_{k+1} + (1 - K)(x_k^{opt} + u_k)$$

K is kalman coefficient and depends of iteration state. We must choose the Kalman coefficient so that the resulting optimal value of the coordinate x_{k+1}^{opt} is as close as possible to the true coordinate x_{k+1}. For example, if we know that our sensor is very accurate, then we will trust its readings more and give the value z_{k+1} more weight (K close to one). If the sensor, on the contrary, is not at all accurate, then we will focus more on the theoretically predicted value $x_{k+1}^{opt} + u_k$. In general, to find the exact value of the Kalman coefficient , you just need to minimize the error:

$$E(e_{k+1}^2) \to min$$

2.3 Generative Adversarial Networks (GANs)

2.3.1 Observation of GANs

Generative Adversarial Networks(GAN) it is a new modern way to data imputaion. The main GAN working concept is minmax game between Generator(G) and Discriminator(D). The generator attempts to produce synthetic data that is indistinguishable from the real data, while the discriminator aims to correctly distinguish between real and generated (or imputed) samples.[2]

$$\min_G \max_D V(D, G) = E_x[logD(x)] + E_z[log(1D(G(z)))]$$

This setup encourages the generator to impute missing values in such a way that the discriminator cannot tell the difference between observed and imputed entries. Until 2018, GAN modelse used to generate images[3], but in 2018 Jinsung Yoon, James Jordon and Mihaela van der Schaar publish Generative Adversarial Imputation Nets one of the earliest and most influential applications of GANs specifically designed for missing data imputation. In 2019 Eoin Brophy, Zhengwei Wang, Qi She, Tomas Ward present Time-series Generative Adversarial Networks.

[2]Goodfellow et al. (2014)

[3]Sajjadi, Mehdi S. M.; Schölkopf, Bernhard; Hirsch, Michael (December 23, 2016). "EnhanceNet: Single Image Super-Resolution Through Automated Texture Synthesis"

2.3.2 GAIN and TimeGAN for Missing Data Imputation

Rather than generating entirely new samples from noise, GAIN is tailored to complete partially observed data. It does so by leveraging adversarial learning to produce realistic estimates of missing entries, effectively mimicking the statistical properties of the observed data. In imputation tasks, particularly in GAIN or time-series-aware GANs, the generator is conditioned on incomplete data and a mask matrix M0,1 n , indicating which values are missing (0) or observed (1). The objective function is adjusted accordingly:

$$\min_G \max_D V(D,G) = E_{x,M} \sim [logD(x,M)] + E_{\hat{x}\sim G(x,M)}[log(1D(G(\hat{x},M)))]$$

This setup encourages the generator to impute missing values in such a way that the discriminator cannot tell the difference between observed and imputed entries. TimeGAN extends the GAN architecture to address the unique challenges of time series generation and imputation, including temporal consistency and dynamics. Unlike GAIN, which treats data as independent rows. In financial time series, missing data often exhibits patterns that are temporally correlated, such as gaps during market closures or due to data transmission failures. Methods that fail to incorporate temporal structure may yield imputations that violate underlying dynamics such as trends, seasonality, or volatility clustering. TimeGAN is uniquely positioned to resolve this by learning both the sequential dependencies and data distribution simultaneously. TimeGAN consists of the following components:

Embedding Network E Maps raw time series XR T×d into a lower-dimensional latent space

$$H = E(X)$$

Recovery Network R Reconstructs the input time series from the latent space:

$$\hat{X} = R(H)$$

Generator G Generates synthetic latent representations $\hat{H}$ from noise vectors $Z \sim N(0, I)$ and optionally past values.

Discriminator D Distinguishes between real and generated latent sequences.

Supervisor S Predicts the next step in the latent space to enforce temporal consistency.

$$\hat{H}_{t+1} = S(H_t)$$

TimeGAN minimizes a weighted sum of four loss components:

$$L_{Total} = \lambda_1 L_{recon} + \lambda_2 L_{adv} + \lambda_3 L_{sup} + \lambda_4 L_{pred}$$

3 Methodology

3.1 Dataset Description

The dataset used in this study consists of daily closing prices for the stock of Sberbank of Russia (SBER) over the course of 12 months in 2017. The data was obtained from the official website of the Moscow Exchange (MOEX). After removing non-trading days and holidays, the dataset includes 252 trading days, represented as a univariate time series.

3.2 Simulating Missing Data

To evaluate imputation performance under controlled conditions, we introduced artificial missingness by randomly removing 20% of the values in the time series. The missing values were generated according to the Missing Completely at Random (MCAR) mechanism, ensuring that the probability of a value being missing is independent of both observed and unobserved data.

3.3 Imputation Methods

To compare the effectiveness of different approaches, we applied the following imputation techniques:

Kalman Filter (KF): A state-space model that recursively estimates missing values based on prior and current state estimates. Suitable for capturing time dependencies in linear and Gaussian settings.

Moving Average (MA): A simple smoothing technique where each missing value is replaced by the mean of its neighboring observed values.

Interpolation: Interpolation techniques estimate missing values by constructing new data points within the range of a discrete set of known values.

They are widely used due to their simplicity and low computational cost. In this study, the following interpolation methods were applied: Linear Interpolation, Polynomial Interpolation, Spline Interpolation:

Generative Adversarial Imputation Networks (GAIN): A GAN-based method designed for missing data imputation using adversarial training and a hint mechanism to guide the discriminator. In the model we can change size of baches or number of iteration, which will make our model more complicate and increase time of working and energy consuming.

TimeGAN: A GAN-based model tailored for time series. It combines adversarial training with supervised learning in a latent space, enabling it to preserve both feature-level realism and temporal dependencies. To evaluate the performance of deep generative models for imputation, we implemented a custom version of TimeGAN, specifically adapted for univariate financial time series with missing values. The model consisted of five key modules: Embedder, Recovery, Generator, Supervisor, and Discriminator, each built using stacked GRU layers. The configuration was as follows:

Sequence length $(seq_len) = 24$

Number of variables $(n_seq) = 1$ (using only CLOSE price)

Hidden dimension $(hidden_dim) = 24$

Optimizer: Adam with learning rate 0.0005

A binary mask vector was used to track the locations of missing values, which was later used in GAN-based models such as GAIN and TimeGAN.

3.4 Metrics

Metrics To measure imputation accuracy, we used the Root Mean Squared Error (RMSE) between the true and imputed values at the missing positions:

$$RMSE = \frac{1}{N_{miss}} \sum_{i \in missing} (x_i - \hat{x}_i)^2$$

. Where: x_i: original (true) value at position, $\hat{x}_i$: imputed value, N_{miss}: total number of missing values. All models were evaluated on the same missing data mask to ensure comparability. Each experiment was run multiple times with different random seeds, and the average RMSE was reported.

4 Evaluation

4.1 interpolation

To assess the performance of basic statistical imputation methods, we applied three types of interpolation—linear, spline, and polynomial—to the Sberbank stock price time series with 20% of values randomly removed (MCAR). The plot below visualizes the comparison between the original data, missing points, and each interpolation method: As shown, all three methods provide

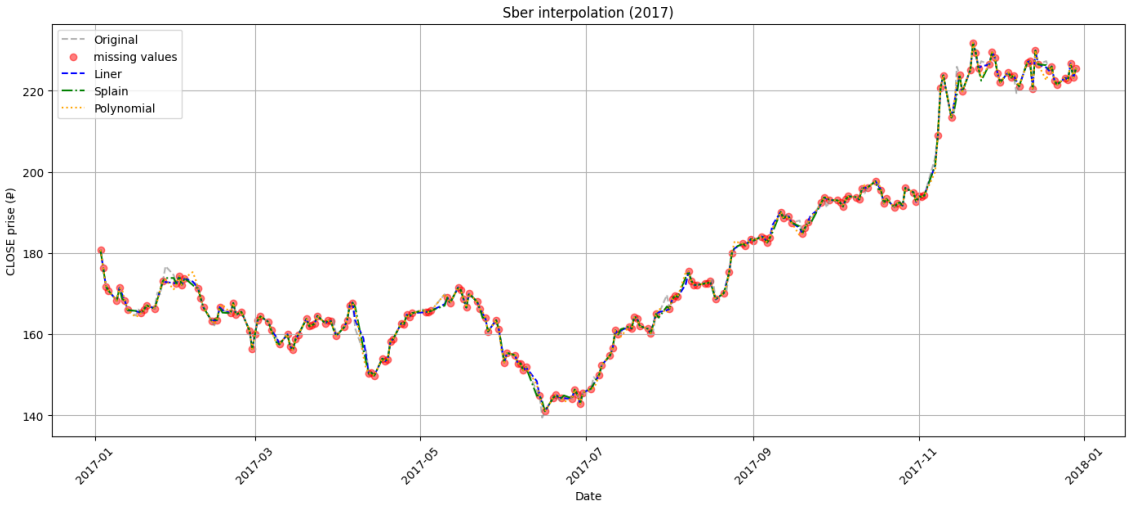

Figure 1: interpolation imputation

visually plausible reconstructions of the missing values, but slight deviations can be observed, especially during periods of high volatility.

Method	RMSE (↓)
Linear Interpolation	1.88
Spline Interpolation	2.09
Polynomial Interpolation	2.30

Table 1: RMSE for Interpolation

The linear interpolation method achieved the lowest RMSE, suggesting it performed best in approximating the original data under the given missing pattern. Surprisingly, spline and polynomial interpolations introduced

slightly larger errors, possibly due to overfitting local patterns or sharp changes in the time series.

4.2 Kalman Filter

The Kalman Filter was applied as a model-based approach for imputation, treating the stock price as a latent process observed through noisy measurements. It recursively estimates missing values based on prior observations and estimated state transitions. The figure below shows the original Sber-

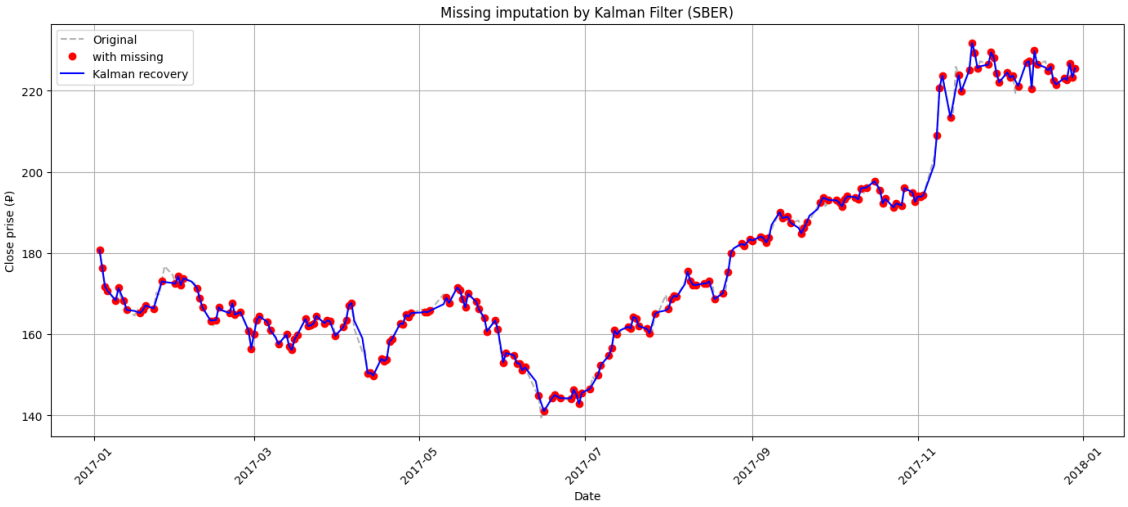

Figure 2: Kalman Filter imputation

bank closing prices (in gray), the time series with 20% of values randomly removed (in red), and the reconstructed values using the Kalman filter (in blue):

While the Kalman filter is theoretically well-suited for time series data, its performance is affected by strong assumptions of linearity and Gaussian noise, which may not fully reflect the dynamics of financial markets. The Kalman filter performs similarly to linear interpolation in terms of RMSE, though its performance may vary depending on market volatility. In some segments (e.g. late 2017), the reconstruction shows visible overshooting or lagging, which may explain its limitations under non-stationary conditions.

4.3 Rolling mean and ARMA

The visualization below shows the imputed values using Rolling Mean (green dashed line) and ARMA (blue solid line), alongside the original data (gray) and missing values (red dots):

Despite its theoretical potential, ARMA failed to produce any imputed values in the masked regions, returning only values for already observed points. This may be due to limitations in model initialization or failure to converge under sparse inputs.

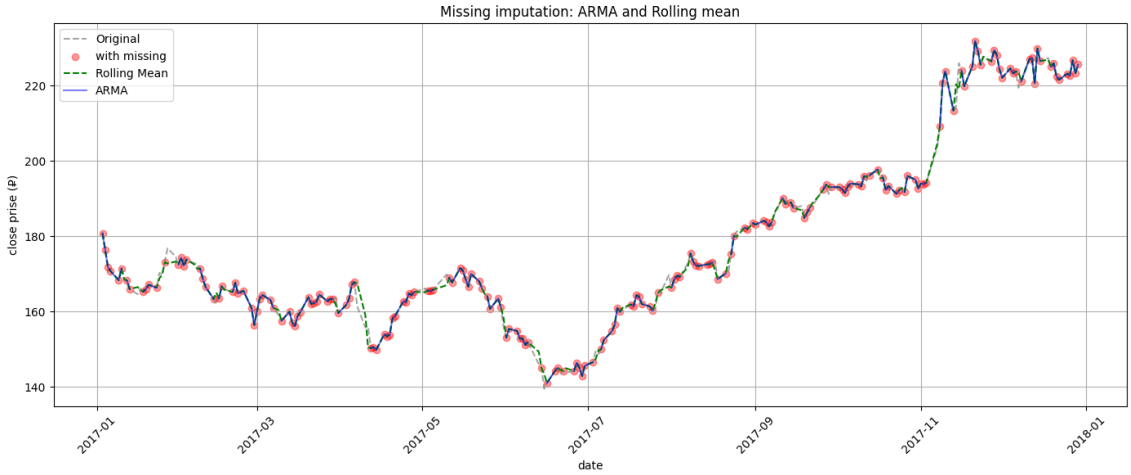

Figure 3: Rolling mean and ARMA imputation

In contrast, Rolling Mean successfully imputed approximately 50 missing values and achieved a moderate reconstruction quality.

The Rolling Mean method, while simple and effective for smoothing, is not designed to model temporal patterns or trends, which likely explains its relatively high RMSE. ARMA, despite its frequent use in time series forecasting, proved unsuitable for direct imputation without further tuning or adaptations for missing data handling. The GAIN model, based on the GAN framework, was applied to impute missing values in the Sberbank 2017 stock price time series. Unlike interpolation and statistical methods, GAIN learns to model the underlying data distribution by training a generator to fill in missing values and a discriminator to distinguish real from imputed data. It uses a hint vector to guide learning and improve stability.

4.4 GAIN

The plot below shows the original data (black), the time series with 20% missing values (dashed red), and the GAIN-imputed series (green) Despite its theoretical advantages, GAIN produced erratic and unstable imputed values, many of which deviated significantly from the original scale (e.g., dropping to zero or near-zero values). This suggests that the model failed to converge properly or did not adapt well to the characteristics of the financial time series. The RMSE of 21.76 on missing values is an order of magni-

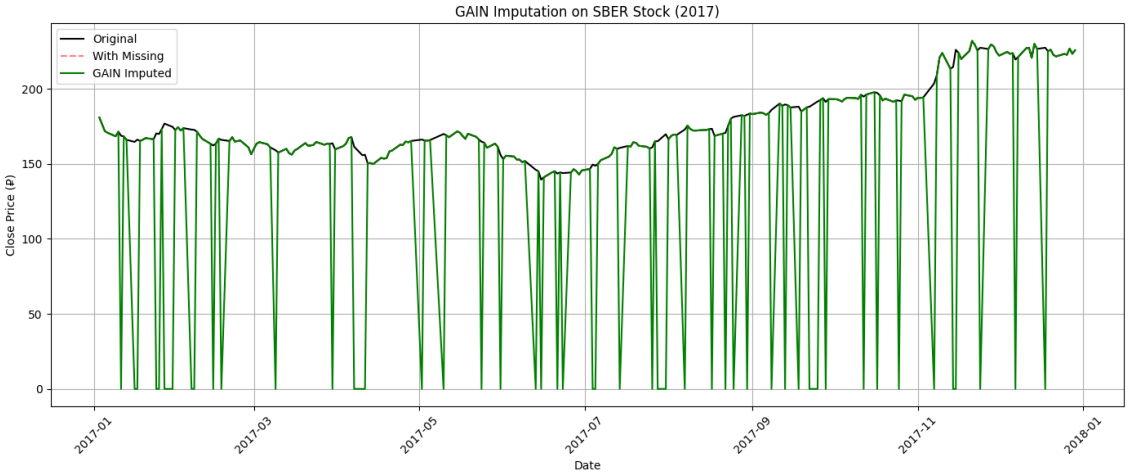

Figure 4: GAIN imputation

tude worse than all other methods tested, indicating poor performance. Any changes in the model like batch size increasing or increasing number of iteration Potential causes may include: Failure to normalize inputs and outputs properly, Inadequate hyperparameter tuning, Sensitivity to time-series non-stationarity and volatility,

4.5 TimeGAN

TimeGAN is a generative adversarial model specifically designed for time series data. It combines supervised sequence modeling with adversarial training, and includes an embedding network that maps data to a lower-dimensional latent space where temporal dependencies are preserved.

The figure below shows the original series (blue), missing data points (red dotted), and values imputed by TimeGAN (green) Despite its tailored architecture, TimeGAN performed poorly on this univariate financial dataset in the imputation task. The imputed values exhibit extreme jumps and frequent divergence from the underlying time series, particularly during high-volatility periods. This indicates possible overfitting, instability during training, or failure to effectively learn short-term dependencies from limited data.

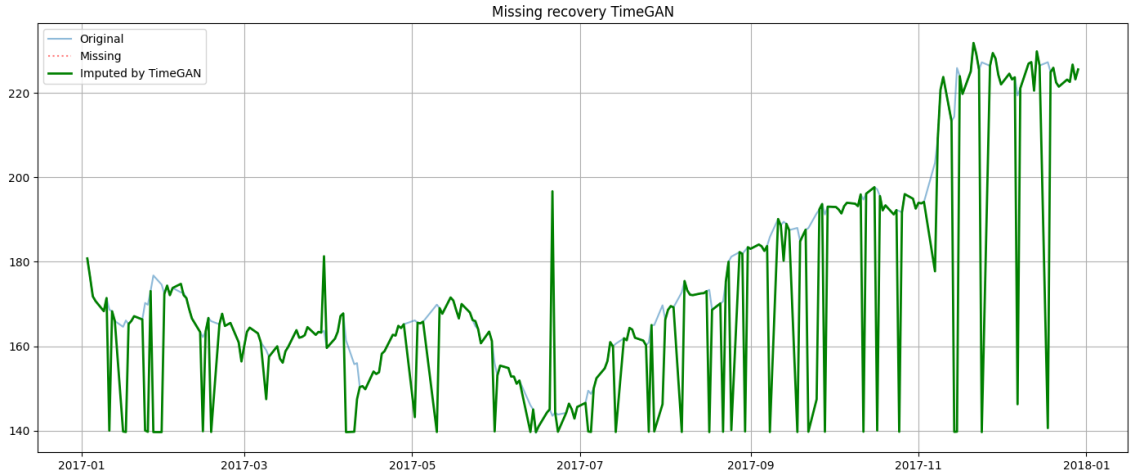

Figure 5: TimeGAN imputation

This is the worst RMSE among all tested methods, indicating that the model did not generalize well to this small-scale imputation task, even though it has shown promising results in data generation and augmentation contexts in other domains. The training was conducted in phases. First, the Embedder and Recovery networks were trained to minimize the reconstruction loss:

$$L_{recon} = E[\|X - \hat{X}\|^2]$$

This phase converged successfully and produced low training MSE. However, due to limited data (252 daily stock prices), full adversarial training (i.e., Generator vs. Discriminator) and supervised loss optimization were not included, which are typically essential in TimeGAN's full pipeline.

Imputation was carried out by:

Filling NaNs with zeros (while tracking a binary mask).

Normalizing the data using MinMaxScaler.

Creating overlapping windows of length 24 for input into the Embedder.

Passing latent representations through the Recovery network.

Reconstructing the full time series and inverting the scaling.

However, as shown in the imputation plot and confirmed by the high RMSE (39.33), the imputed values deviated significantly from the true price series. The recovered data exhibited: Frequent sharp discontinuities,Unrealistic spikes or drops, Lack of temporal smoothness, especially in periods of volatility

5 Results

The performance of each imputation method was assessed using Root Mean Squared Error (RMSE), calculated only at the artificially introduced missing values (20% MCAR). The table below summarizes the results in Table 2.

Method	RMSE (↓)
Linear Interpolation	1.88
Kalman Filter	1.88
Spline Interpolation	2.09
Polynomial Interpolation	2.30
Rolling Mean	2.49
GAIN	21.76
TimeGAN	39.33
ARMA	N/A

Table 2: RMSE comparing

These results show that simpler statistical methods consistently outperformed deep learning models on this task. Both linear interpolation and Kalman filter achieved the lowest RMSE (1.88), followed by spline and polynomial interpolation. Surprisingly, the deep generative models (GAIN and TimeGAN) produced much higher RMSE values, failing to capture the structure of the financial time series accurately. Plots of the reconstructed time series (Figures 1–5) further support the quantitative findings: Interpolation methods produced smooth, stable imputed values closely following the original trend. The Kalman filter provided dynamic estimates that adjusted well to local trends, although occasionally overreacted to volatility.

GAIN and TimeGAN both generated unstable or extreme values, with TimeGAN producing sharp spikes and unrealistic transitions, especially in high-volatility periods. Both GAIN and TimeGAN struggled particularly during volatile periods (e.g., late 2017 price surge), where even small errors in temporal dynamics led to significant deviations. Simpler models were more robust in these regions due to their deterministic, constrained nature.

6 Conclusion

This study investigated the effectiveness of various imputation techniques on financial time series data, with a particular focus on generative adversarial network-based models such as GAIN and TimeGAN. Using daily closing prices of Sberbank stock from 2017, we simulated a missing data scenario with 20

The results clearly show that simpler, statistical methods like linear interpolation and the Kalman filter outperformed more complex models in this specific setting. Both achieved the lowest RMSE of 1.88, followed by spline and polynomial interpolation. In contrast, the deep learning models — GAIN and especially TimeGAN — performed significantly worse, with RMSE values of 21.76 and 39.33 respectively.

The poor performance of GAIN and TimeGAN can be explained by several factors. First, these models require substantial data and careful tuning, which are difficult to achieve with a short, univariate dataset like ours. Second, while these models are theoretically powerful, their full potential often emerges in higher-dimensional or multivariate settings. In our implementation of TimeGAN, only the autoencoder phase was fully trained due to data and computational constraints, which likely limited its ability to learn temporal dependencies and produce realistic imputations.

Despite this, the experiment demonstrates the value of generative models as a research direction. Even though the results here were not favorable to GANs, the framework allows for flexible extensions, such as incorporating domain knowledge, conditioning on external variables, or learning latent structure. With more data and computational resources, these models may outperform traditional methods in more complex real-world scenarios.

In conclusion, when working with limited financial data, simple imputation techniques remain highly competitive, often outperforming sophisticated deep learning models in both accuracy and reliability. However, GAN-based

models still hold promise and deserve further exploration, particularly in settings with richer datasets and more complex temporal structure.

7 References

References

[1] Goodfellow, I., et al. (2014). *Generative adversarial nets*. Advances in Neural Information Processing Systems, 27.

[2] Yoon, J., Jordon, J., & van der Schaar, M. (2018). *GAIN: Missing data imputation using generative adversarial nets*. arXiv preprint arXiv:1806.02920.

[3] Yoon, J., Jarrett, D., & van der Schaar, M. (2019). *Time-series Generative Adversarial Networks*. NeurIPS.

[4] Esteban, C., Hyland, S. L., & Rätsch, G. (2017). *Real-valued (medical) time series generation with recurrent conditional GANs*. arXiv preprint arXiv:1706.02633.

[5] Ramponi, G., Protopapas, P., Brambilla, M., & Janssen, R. (2018). *T-CGAN: Conditional generative adversarial network for data augmentation in noisy time series with irregular sampling*. arXiv preprint arXiv:1811.08295.

[6] van der Maaten, L., & Hinton, G. (2008). *Visualizing data using t-SNE*. Journal of Machine Learning Research, 9(Nov), 2579–2605.

[7] Bryant, F. B., & Yarnold, P. R. (1995). *Principal-components analysis and exploratory and confirmatory factor analysis*.

[8] Yoon, J., Jordon, J., & van der Schaar, M. (2019). *PATE-GAN: Generating synthetic data with differential privacy guarantees*. International Conference on Learning Representations (ICLR).

[9] Williams, R. J., & Zipser, D. (1989). *A learning algorithm for continually running fully recurrent neural networks*. Neural Computation, 1(2), 270–280.

[10] Bengio, Y., Louradour, J., Collobert, R., & Weston, J. (2009). *Curriculum learning*. Proceedings of the 26th ICML, 41–48.

[11] Ganin, Y., et al. (2016). *Domain-adversarial training of neural networks*. Journal of Machine Learning Research, 17(1), 2096–2030.

[12] Konda, V. R., & Tsitsiklis, J. N. (2000). *Actor-critic algorithms*. In Advances in Neural Information Processing Systems, 1008–1014.

[13] Mirza, M., & Osindero, S. (2014). *Conditional generative adversarial nets*. arXiv preprint arXiv:1411.1784.

[14] Zhang, Y., Gan, Z., & Carin, L. (2016). *Generating text via adversarial training*. NIPS workshop on Adversarial Training.

[15] Simonetto, L. (2018). *Generating spiking time series with GANs: an application on banking transactions*.

[16] Haradal, S., Hayashi, H., & Uchida, S. (2018). *Biosignal data augmentation based on GANs*. EMBC 2018: 40th Annual Intl. Conf. of the IEEE Engineering in Medicine and Biology Society, 368–371.

[17] Alzantot, M., Chakraborty, S., & Srivastava, M. (2017). *SenseGen: A deep learning architecture for synthetic sensor data generation*. IEEE PerCom Workshops.

[18] Zhang, C., Kuppannagari, S. R., Kannan, R., & Prasanna, V. K. (2018). *GAN for synthetic time series in smart grids*. IEEE SmartGridComm.

[19] Chen, Y., Wang, Y., Kirschen, D., & Zhang, B. (2018). *Model-free renewable scenario generation using GANs*. IEEE Transactions on Power Systems, 33(3), 3265–3275.

[20] Dai, A. M., & Le, Q. V. (2015). *Semi-supervised sequence learning*. In NeurIPS, 3079–3087.

[21] Lyu, X., Hueser, M., Hyland, S. L., Zerveas, G., & Rätsch, G. (2018). *Improving clinical predictions through unsupervised time series representation learning*. arXiv preprint arXiv:1812.00490.

[22] Srivastava, N., Mansimov, E., & Salakhutdinov, R. (2015). *Unsupervised learning of video representations using LSTMs*. In ICML, 843–852.

[23] Fabius, O., & van Amersfoort, J. R. (2014). *Variational recurrent auto-encoders*. arXiv preprint arXiv:1412.6581.

[24] Li, Y., & Mandt, S. (2018). *Disentangled sequential autoencoder*. arXiv preprint arXiv:1803.02991.