MONASH University

Combining Full Bayesian Inference with Prior-fitted Networks for Time Series Forecasting

Che-Yu Wu
Student ID: 31977251
Course: FIT5128

Supervisor: Dr. Angus Dempster
Co-supervisors: Dr. Christoph Bergmeir
Prof. Daniel Schmidt

Word Count: 7879

Acknowledgements

I would like to gratefully thank my supervisors for their effort throughout this journey. No matter the circumstances, they are always patient and never hesitant to help and share their knowledge. Dr. Angus Dempster is always calm, cool, and collected. He always brings a very positive attitude and is willing to help me whenever I need. He is not only a knowledgable supervisor, but also a mentor that I have looked up to. Dr. Christoph Bergmeir and Prof. Daniel Schmidt are also very kind and willing to share a lot of knowledge with me, their guidance really helped me a lot when I am lost. Without the help of my supervisors, I cannot complete this work and my research. Their support has been invaluable. I wish them all the best in their future, especially my primary supervisor Angus, his guidance has meant a lot to me throughout the year.

Table of Contents

I Literature Review (Resubmission) 1

1 Introduction 1

2 Literature Review 3

2.1 Time Series Forecasting 3

2.1.1 Time Series 3

2.1.2 Time Series Forecasting 4

2.2 Classical Time Series Forecasting Methods 4

2.2.1 The ARIMA Method 4

2.2.2 The ETS Method 5

2.3 Bayesian Approach and Neural Networks 6

2.3.1 Bayesian Approaches to Time Series 6

2.3.2 Neural Networks for Timeseries Forecasting 7

2.3.3 Incorporating Bayesian Methods with Neural Networks . 8

2.4 Prior-Fitted Networks 9

2.4.1 Foundations and Meta-Learning Framework 9

2.4.2 Statistical Foundation 10

2.4.3 Application to Forecasting: ForecastPFN 10

2.4.4 TabPFN and TabPFN-TS: Generalisation Beyond Forecasting 10

2.4.5 Advantages and Limitations 11

3 Summary of the State of Art **11**

4 Research Project Plan **13**

4.1 Objective 13

4.2 Methodology 13

4.2.1 Model Selection: Bayesian Time Series Models 13

4.2.2 Synthetic Prior Construction 14

4.2.3 Prior-Fitted Network Training 14

4.2.4 Forecasting Benchmarks and Evaluation 15

4.2.5 Forecasting Tasks 16

4.3 Data Sources 16

4.4 Timeline and Milestones 17

4.5 Ethics 17

5 Conclusion **17**

6 Reference List **19**

II Research Paper: Combining Full Bayesian Inference with Prior-fitted Networks for Time Series Forecasting **26**

1 Introduction **26**

2 Background **28**

2.1 Time Series Forecasting 28

2.2 Prior-fitted Networks 28

2.2.1 Theoretical Foundation 29

2.2.2 Properties and Advantages 30

2.2.3 Different Types of Priors and Applications 31
2.3 Full Bayesian Inference & LGT model 31
2.3.1 Introduction to Full Bayesian Inference in Time Series . . 31
2.3.2 The LGT Model Structure 32
2.3.3 LGT's Suitability as a Prior for PFN 32
2.4 Time Series Forecasting Evaluation 32
2.4.1 The Problem with Scale-Dependent Metrics 32
2.4.2 Scale-Independent Alternatives 33

3 Methodology **34**

3.1 Using LGT as a Prior for Synthetic Data Generation 34
3.1.1 From Forecasting Model to Synthetic Data Generator . . 34
3.1.2 Simulation Process 34
3.1.3 Using M3 Monthly Data for Learning Parameter Distributions 35
3.2 Neural Network Architecture 36
3.2.1 ForecastPFN Overview 36
3.2.2 Architecture Components 37
3.2.3 Implementation Issues and Corrections 39
3.2.4 Input and Output Format 40
3.3 Training Details 40
3.3.1 Training Data and Preparation 40
3.3.2 Loss Function and Optimisation 41
3.3.3 Training Configuration 41
3.4 Inference and Prediction Process 41

4 Results and Discussion **42**

4.1 Setup 42
4.1.1 Datasets 42
4.1.2 Baseline Methods 42
4.1.3 Evaluation Metrics 43
4.2 Main Results 43
4.2.1 Overall Performance Patterns 44
4.2.2 M3 Monthly and Yearly Performance 45
4.2.3 Monash Repository Datasets Performance 46
4.3 Interpretation, Limitations, and Implications 47
4.3.1 ForecastPFN Implementation Issues: The Primary Confounding Factor 47
4.3.2 Additional Limiting Factors 48
4.3.3 Implications and Partial Successes 48
4.4 Future Directions 49

5 Conclusion **49**

6 Ethics Statement **50**

7 References **51**

Appendix **55**

A.1 Sample Series 55
A.2 M3 Monthly Prediction Examples 55
A.3 M3 Yearly Prediction Examples 56

Part I

Literature Review (Resubmission)

1 Introduction

Time series forecasting plays a vital role in domains such as business planning, stock market prediction, weather forecasting, and energy management [1]. Its integration into decision-making processes is essential across many fields [2]. Traditional statistical methods like Auto Regressive Integrated Moving Average (ARIMA) and Exponential Somoothing (ETS) have long provided interpretable and theoretically grounded forecasting tools. However, the rise of machine learning, particularly deep learning, has transformed the field by enabling the modelling of complex non-linear patterns. In the M5 competition, all 50 top-performing methods were machine learning-based, with deep learning approaches showing potential for even greater accuracy, especially with hierarchical data [3]. Despite these advancements, challenges remain in achieving consistent, high-quality forecasts across varied real-world scenarios [4].

Statistical time series forecasting models provide a well-grounded theoretical framework for forecasting, with clearly defined assumptions and interpretable parameters. These models perform well when it comes to capturing temporal dependencies like seasonality, trends, and autoregressive patterns that are common in many real-world time series [5]. However, these models often rely on simplified assumptions about the data generation process, such as linearity and Gaussian error distributions [5], which may not reflect real-world complexity. Furthermore, these models can only effectively handle short sequences and perform poorly when faced with the complex sequences common in today's applications [6].

On the other hand, deep learning neural network methods showcases advantages like forecasting without assumptions or pre-defined data generating process [7]. For instance, Long Short-Term Memory (LSTM) networks [8] are capable of learning complex temporal dependencies directly from raw sequential data, making them effective in capturing both short- and long-term patterns in time series without manual feature engineering or model specification. The M4 competition marked a turning point, showing that classical statistical methods no longer consistently outperformed more complex models [3]. Makridakis et al. [9] demonstrated that hybrid approaches, combining neural networks with statistical methods, achieved superior forecasting performance. This trend continued in the M5 competition, where machine learning-based methods dominated

[3]. Deep learning models, however, have the black-box nature which makes the models hard to interpret and explain by researchers [10], [11]. This limitation reduces their utility in scenarios where objects must provide explanations for their decision-making processes [12].

Bayesian inference is a statistical method that utilises Bayes' theorem to update probabilities as new data becomes available [13]. It provides a principled framework for incorporating prior knowledge and quantifying uncertainty, which is especially valuable in forecasting. For example, Bayesian ARIMA models extend the classical ARIMA framework by placing priors over parameters, enabling posterior predictive intervals. While this improves uncertainty quantification, it also increases computational demands, particularly when using sampling-based methods like Markov Chain Monte Carlo (MCMC) [14]. The Local and Global Trend (LGT) model by Smyl et al. [15] illustrates these trade-offs: it uses the No-U-Turn Sampler (NUTS) in Stan to generate accurate forecasts with robust uncertainty estimates but remains computationally intensive and less scalable to high-frequency or long time series. These challenges have historically limited broader adoption of Bayesian forecasting methods [16]. Nevertheless, as computational resources have improved in recent decades, the use of Bayesian inference has expanded steadily, allowing researchers to explore its potential more widely [14].

The concept of Prior-Fitted Networks (PFNs), recently introduced by Müller et al. [17], represents a promising approach to bridging the gap between classical statistical models and modern neural networks. PFNs are designed to approximate Bayesian posterior predictive distributions through synthetic meta-learning, enabling models to generalise rapidly to new datasets without retraining [17]. By learning from entire synthetic datasets generated from a prior, PFNs can perform zero-shot forecasting, producing predictions and well-calibrated uncertainty estimates in a single forward pass [18]. This method preserves the probabilistic foundations of Bayesian inference while leveraging the computational speed and flexibility of deep learning architectures, particularly Transformers. Although PFNs have demonstrated strong empirical performance in areas such as tabular classification and time series forecasting [18], [19], [20], their application remains at an early stage. Notably, models like TabPFN-TS have been trained using relatively generic synthetic priors, without fully exploiting the structured models developed in Bayesian forecasting. While the results achieved by TabPFN-TS are promising, they suggest that there may be substantial room for improvement by designing priors based on more realistic Bayesian time series structures, such as Bayesian ARIMA or ETS models. This apparent gap between simple synthetic training and the rich modelling capabilities available in Bayesian statistics provides a clear motivation for the present research.

There has been relatively little research combining Bayesian methods with neural networks for time series forecasting. This review explores their intersec-

tion, with a focus on Bayesian statistical approaches. It examines how these methods have developed independently and highlights early attempts at integration, identifying key strengths, limitations, and research gaps. By assessing uncertainty quantification, computational efficiency, and predictive accuracy, this review lays the groundwork for a proposed framework that combines full Bayesian inference with prior-fitted networks. Addressing these gaps could lead to more robust and interpretable forecasting tools for real-world decision-making.

2 Literature Review

The field of time series forecasting has evolved significantly over the past decades, progressing from interpretable statistical models to probabilistic Bayesian frameworks and more recently to powerful deep learning architectures. Each stage of this evolution has addressed specific limitations of its predecessors, including capturing complex dependencies, representing uncertainty, or scaling to large datasets. This literature review follows this progression, highlighting the conceptual shifts and methodological innovations that have shaped current forecasting practices. Particular attention is given to emerging approaches that seek to unify these traditions, such as prior-fitted networks, which combine Bayesian reasoning with the representational capacity of neural models.

2.1 Time Series Forecasting

2.1.1 Time Series

A time series is mathematically represented as $X = x_1, x_2, ..., x_t$, where each x_t denotes the value of a variable observed at time t [21]. The structure of a time series typically includes several key components. The trend reflects long-term directional movement, while seasonality captures systematic, calendar-based fluctuations that recur at fixed intervals [21]. Cyclic patterns represent broader, non-periodic variations often linked to economic or structural factors [21]. Finally, the irregular or residual component accounts for random noise unexplained by the systematic elements [21]. These components may combine additively or multiplicatively and are commonly separated through decomposition techniques to support effective forecasting model development.

2.1.2 Time Series Forecasting

Time series forecasting is the process of predicting future values of a time-dependent variable based on its historical observations. It is typically formulated as an autoregressive task, where future values $x_{t+1}, x_{t+2}, \ldots$ are predicted using past observations $x_1, x_2, \ldots, x_t$ [21]. Accurate forecasting requires models that can effectively capture the temporal structure of the data, including trend, seasonality, and autocorrelation. Classical methods such as ARIMA and ETS rely on statistical assumptions to model these patterns [5], while modern machine learning and deep learning approaches aim to learn them directly from data [7]. Forecasting plays a crucial role in various fields such as finance, supply chain planning, and environmental monitoring, where anticipating future outcomes supports timely and informed decision-making [1].

2.2 Classical Time Series Forecasting Methods

ARIMA and ETS have long been recognised as two of the most influential classical methods for time series forecasting. Their success lies in their strong theoretical basis, interpretability, and consistent performance across a variety of practical applications. Even today, they continue to be widely applied in forecasting tasks. However, as the complexity of real-world datasets grows, these methods encounter notable challenges, particularly in handling non-linear patterns and irregular structures. This section examines the strengths that have sustained the relevance of ARIMA and ETS, while also highlighting their limitations in addressing the demands of modern forecasting problems [5].

2.2.1 The ARIMA Method

The Auto Regressive Integrated Moving Average (ARIMA) method was first introduced by Box and Jenkins in 1970 [22]. The book "Time Series Analysis: Forecasting and Control", which they published, has established a systematic framework for analysing and forecasting time series data. Since its first publication, this book has continued to serve as the basis for statistical forecasting. The ARIMA model, just like its name, is a model that integrates autoregression (AR) and moving average (MA). Compared to the notation ARMA(p,q), the notation ARIMA(p,d,q) is formed by "p" autoregressive terms, "q" moving average terms, and a new term "d" which represents level of differencing [22]. In general, as stated by Shumway and Stoffer [23], the mathematical formulation can be written as:

$$\phi(B)(1-B)^d x_t = \theta(B) w_t$$

In order to effectively develop and implement ARIMA models, the Box-

Jenkins methodology is commonly applied. This approach begins by using the autocorrelation function (ACF) and partial autocorrelation function (PACF) to help identify the most suitable values for the parameters p, d, and q [24]. If the plots are not conclusive, model selection criteria such as the Bayesian Information Criterion (BIC) and Akaike Information Criterion (AIC) are used instead [24]. The model with the lowest BIC or AIC value is considered the best fit [24].

The ARIMA model has successfully demonstrated its capability in fields such as economics, finance, and environmental science [25], [26]. As a well-established model, it has advantages such as interpretability, existing statistical properties, and sufficient software support [23], [27]. However, the ARIMA model has some significant limitations. It assumes linear relationships, struggles with complex seasonality patterns, and provides limited information about prediction uncertainty [28], [29], [30]. Furthermore, traditional ARIMA models also cannot easily incorporate exogenous variables, leading to the development of extensions such as ARIMAX and SARIMAX models [31], [32]. Despite these limitations, ARIMA models continue to serve as both practical forecasting tools and important baselines against which more complex models are compared.

2.2.2 The ETS Method

The exponential smoothing method (ETS) is invented by Robert G. Brown during World War II when he was an operational research analyst for the US Navy in 1950 [33]. Despite this early introduction, the rigorous statistical foundation has not been stated until the proposal of the state space formulation by Hyndman et al. in 2002 and 2008 [34], [35]. As stated by Hyndman et al. [35], the ETS can be decomposed into error (E), trend (T), and seasonal (S) components, with each component generally specified as additive (A), multiplicative (M), or none (N) [35]. For instance, a basic additive model ETS(A,A,N) can be written as:

$$y_t = l_{t-1} + b_{t-1} + \epsilon_t$$
$$l_t = l_{t-1} + b_{t-1} + \alpha\epsilon_t$$
$$b_t = b_{t-1} + \beta\epsilon_t$$

Where y_t is the observation, l_t is the level, b_t is the trend, and α and β are smoothing parameters.

However, ETS methods still face important challenges. For example, the way the model assigns weights to past observations can sometimes be inappropriate, leading to biased forecasts. There are also difficulties in setting the initial values for model parameters, which can impact the stability and accuracy of predictions. Additionally, because ETS models use fixed weighting over time, they may struggle to adapt to sudden changes or shifts in the underlying data, limiting their performance in more dynamic forecasting environments [36].

2.3 Bayesian Approach and Neural Networks

2.3.1 Bayesian Approaches to Time Series

Bayesian methods offer a principled framework for time series forecasting by treating model parameters as random variables rather than fixed quantities. This allows for the integration of prior knowledge and explicit modelling of both parameter uncertainty (the variability in estimated model coefficients) and forecast uncertainty (the variability in future predictions). For example, instead of providing only a point estimate for next month's sales, a Bayesian ARIMA model can generate a predictive interval, such as predicting that sales will likely fall between 950 and 1,050 units with 95% probability, reflecting both estimation error and inherent randomness in the process.

The foundation for Bayesian time series modelling was laid by West and Stevens in 1976 [37] and later formalised by West and Harrison in their influential work "Bayesian Forecasting and Dynamic Models" [38]. However, for many years, practical application was constrained by computational intractability. Full Bayesian inference relies heavily on Markov Chain Monte Carlo (MCMC) techniques, such as Gibbs sampling [39] and the Metropolis-Hastings algorithm [40]. Although robust, they require generating thousands to millions of posterior samples to approximate distributions accurately. This makes MCMC orders of magnitude slower than classical maximum likelihood estimation or typical neural network training, particularly for high-dimensional or large datasets.

Building on these advances, researchers developed Bayesian versions of classical models, such as Bayesian ARIMA [41] and Bayesian Exponential Smoothing [42], which enhance robustness by incorporating prior information and providing credible intervals around forecasts. Bayesian ETS models have proven especially valuable in volatile settings, offering realistic uncertainty quantification where traditional point forecasts would fail.

More recently, the Local and Global Trend (LGT) model proposed by Smyl et al. [14] generalised ETS models by capturing nonlinear growth dynamics and modeling residuals with a Student t-distribution to handle heavy-tailed errors. Despite its accuracy, the LGT model originally relied on the computationally expensive No-U-Turn Sampler (NUTS) in Stan, limiting scalability for larger datasets. Long et al. [43] addressed this issue by introducing a fast Gibbs sampling algorithm for the LGT model. Their unified Local-Seasonal-Global Trend (LSGT) framework merged seasonal and non-seasonal variants and improved sampling efficiency through conditional conjugacy techniques and regularisation with horseshoe priors.

Despite these improvements, Bayesian time series models still face scalability challenges. Full MCMC remains significantly more computationally inten-

sive compared to point estimation methods or deep learning models optimised via stochastic gradient descent. This motivates ongoing exploration of approximate inference methods, such as variational inference [44] and integrated nested Laplace approximations (INLA) [45], which offer faster but approximate posterior estimates. Nevertheless, recent innovations demonstrate that carefully designed Bayesian models can now deliver both accuracy and practical efficiency, broadening their applicability in real-world forecasting tasks.

2.3.2 Neural Networks for Timeseries Forecasting

The emergence of neural networks has introduced powerful alternatives for time series forecasting, enabling the modelling of complex, non-linear relationships beyond the scope of classical statistical methods. Early applications in the late 1980s by Lapedes and Farber [46] and Weigend et al. [47] demonstrated that neural networks could outperform linear models in predicting chaotic time series, laying the foundation for subsequent developments.

Initial efforts focused on Feedforward Neural Networks (FNNs), particularly Multilayer Perceptrons (MLPs), which modelled lagged observations as input features passed through non-linear activation layers. While FNNs effectively approximate non-linear functions, they lack explicit mechanisms for capturing temporal dependencies, treating time merely as an additional input dimension.

The introduction of Recurrent Neural Networks (RNNs) addressed this limitation by allowing models to maintain internal memory states that evolve across time steps. However, standard RNNs struggled with learning long-term dependencies due to the vanishing gradient problem [48]. This challenge was overcome with the development of Long Short-Term Memory (LSTM) networks by Hochreiter and Schmidhuber [8], which introduced memory cells and gating mechanisms to preserve information over longer sequences. Empirical studies have demonstrated the superior forecasting performance of LSTMs in domains such as finance and economics [49], [50], with applications ranging from stock market prediction [51] to macroeconomic forecasting.

Simplified variants such as Gated Recurrent Units (GRUs), introduced by Cho et al. [52], have offered comparable performance with fewer parameters, making them computationally efficient alternatives. Empirical comparisons by Chung et al. [53] suggest that GRUs can match or even outperform LSTMs under certain conditions.

More recently, attention-based architectures like Transformers [54] have been adapted for time series tasks. Models such as the Temporal Fusion Transformer [55] and Informer [56] leverage self-attention mechanisms to dynamically focus on relevant historical data points, achieving state-of-the-art results in multi-horizon forecasting and handling complex dependencies across time scales.

Despite their predictive power, traditional neural networks typically produce point forecasts without native uncertainty quantification, limiting their applicability in risk-sensitive domains. Recent advances, including probabilistic neural networks like DeepAR [57] and deep state space models [58], aim to generate full predictive distributions. Nevertheless, neural models often require large amounts of data, and their black-box nature challenges interpretability—a critical factor in high-stakes forecasting scenarios. Efforts such as attention visualisation and incorporating domain knowledge into neural architectures continue to evolve to address these limitations.

2.3.3 Incorporating Bayesian Methods with Neural Networks

The limitations of traditional Bayesian models and deep learning approaches have motivated the development of hybrid methods that seek to combine the strengths of both methods. Classical Bayesian time series models, while providing uncertainty quantification and theoretical rigor, often struggle with scalability and flexibility when modelling complex, non-linear dynamics. Conversely, neural networks excel at capturing non-linear relationships and handling large datasets but typically lack calibrated uncertainty estimates and are less interpretable.

Hybrid approaches aim to bridge this gap by incorporating Bayesian principles into neural network architectures. One prominent line of research involves Bayesian Neural Networks (BNNs), where uncertainty is introduced by placing probability distributions over network weights rather than using fixed point estimates [59]. In practice, BNNs can quantify uncertainty and reflect it about the model parameters themselves, thus improving forecast credibility in low-data scenarios [59]. However, training full BNNs remains computationally intensive, and scalable approximations such as variational inference are often necessary to make them practical [59].

In time series forecasting, probabilistic deep learning models like DeepAR [57] and Deep State Space Models [58] represent further efforts to integrate predictive uncertainty into deep learning frameworks. DeepAR, for instance, models future sequences as a probability distribution conditioned on past observations, providing full quantile forecasts rather than simple point predictions. Other approaches, such as deep ensembles, train multiple neural networks independently and use their collective predictions to estimate uncertainty, offering a practical compromise between accuracy and computational cost [60].

Recent hybrid developments also focus on combining classical statistical models with deep learning. Methods such as N-BEATS [61] with interpretable blocks, hybrid LSTM-ARIMA models, and domain-informed priors in neural networks reflect ongoing efforts to leverage the strengths of both worlds. These strategies seek to retain the flexibility and predictive strength of neural networks

while embedding structure or prior knowledge that improves interpretability and data efficiency.

A particularly promising innovation in this hybrid space is the emergence of Prior-Fitted Networks [17], which synthesise Bayesian reasoning with the computational advantages of deep learning. PFNs are trained on synthetic datasets generated from a prior distribution, allowing them to approximate Bayesian posterior predictions without requiring online retraining or large labelled datasets. This novel approach offers a pathway to achieving plug-and-play probabilistic forecasting models that are scalable, flexible, and theoretically grounded—addressing many of the challenges that traditional and neural models face individually.

2.4 Prior-Fitted Networks

The recent emergence of Prior-Data Fitted Networks (PFNs) marks a promising shift in the landscape of time series forecasting. Introduced by Müller et al. [15], PFNs are designed to approximate Bayesian posterior predictive distributions through synthetic meta-learning, offering zero-shot prediction and calibrated uncertainty estimates. By training purely on datasets drawn from a prior, PFNs achieve rapid inference without fine-tuning, a major departure from traditional deep learning approaches [19], [62].

2.4.1 Foundations and Meta-Learning Framework

The core innovation behind PFNs lies in their meta-learning approach, which departs from conventional supervised training. Instead of learning from real-world datasets, PFNs are trained exclusively on synthetic datasets sampled from a structured prior [17]. For each synthetic task, the model observes a partial dataset and predicts missing labels, approximating the posterior predictive distribution $p(y \mid x, D)$ directly. This process enables PFNs to generalise rapidly to new tasks without additional fine-tuning, a capability often referred to as in-context learning.

Unlike traditional neural networks that optimise parameters based on observed real-world data, PFNs perform a single forward pass at inference time. Furthermore, PFNs frequently rely on index-based features, such as positional encodings (e.g., time indices), rather than conditioning directly on historical values, contrasting sharply with autoregressive forecasting models. As a result, PFNs implicitly learn a regression mapping between input indices and target outcomes during meta-training, rather than explicitly modelling temporal dependencies. While this enables fast zero-shot inference, it also introduces a potential limitation: if the test data distribution deviates substantially from the

synthetic prior, model generalisation may suffer. The Transformer architecture, with its ability to encode variable-sized datasets via self-attention, provides a natural backbone for implementing PFNs efficiently [19].

2.4.2 Statistical Foundation

PFNs offer an interesting blend of Bayesian and frequentist interpretations. While originally inspired by Bayesian posterior inference, PFNs can also be understood as pre-trained predictors minimising generalisation error through meta-learning [17]. From an information-theoretic view, PFNs aim to make their predictions as close as possible to the true Bayesian posterior by minimising the difference between the two distributions. Importantly, Nagler [62] showed that the variance of PFN predictions decreases as the number of observed context points increases, reflecting desirable statistical properties. Unlike traditional Bayesian methods that often require expensive marginalisation procedures, PFNs directly approximate the posterior predictive distribution in a single forward pass, enabling fast and scalable probabilistic forecasting.

2.4.3 Application to Forecasting: ForecastPFN

ForecastPFN represents the first application of PFNs specifically designed for time series forecasting [19]. Trained entirely on synthetic datasets generated from a structured prior—including seasonality, trends, and noise—ForecastPFN learns generalised temporal patterns without exposure to real-world data. Its key innovation lies in its zero-shot capability: the model can accurately forecast unseen time series without fine-tuning. Benchmark results demonstrate that ForecastPFN outperforms established deep learning models like DeepAR and N-BEATS, even when those models are trained on hundreds of samples. By relying solely on synthetic priors, ForecastPFN also mitigates risks associated with benchmark leakage and dataset-specific overfitting, offering a scalable and interpretable alternative for short and sparse time series forecasting.

2.4.4 TabPFN and TabPFN-TS: Generalisation Beyond Forecasting

Following the success of ForecastPFN, TabPFN extended PFNs to small-scale tabular classification tasks [20]. Trained on synthetic datasets generated from Bayesian neural networks and causal models, TabPFN achieves state-of-the-art performance with minimal inference time and no hyperparameter tuning. Building on this, TabPFN-TS adapts the approach for time series forecasting [18]. Instead of conditioning on past values, temporal information is encoded through calendar-based features, framing time series prediction as a tabular

regression problem. Remarkably, TabPFN-TS matches or outperforms domain-specific models like Chronos-Large and DeepAR, demonstrating that PFN-based architectures can generalise effectively across diverse forecasting tasks without requiring large-scale retraining.

2.4.5 Advantages and Limitations

PFNs offer several distinct advantages for time series forecasting. Their zero-shot capability enables immediate predictions on new datasets without retraining, and their single forward-pass inference dramatically improves computational [62]. PFNs also approximate Bayesian posterior predictive distributions, providing calibrated uncertainty estimates critical for decision-making in sensitive applications [62]. Furthermore, they require no hyperparameter tuning at inference time, offering true plug-and-play deployment. However, PFNs are heavily reliant on the quality of the synthetic prior; poorly specified priors can bias forecasts and limit generalisation [19]. For instance, TabPFN-TS was trained using generic priors derived from tabular regression tasks, rather than structured time series models. While this approach yields surprisingly strong performance, it does not fully exploit domain-specific temporal dynamics, suggesting that the use of Bayesian-informed priors could lead to further gains. Additionally, adapting to new domains or incorporating new priors requires retraining from scratch, which can be computationally costly. Balancing prior design and task specificity remains a key challenge for broader adoption.

3 Summary of the State of Art

Over the past decades, time series forecasting has advanced significantly, progressing from interpretable statistical models like ARIMA and ETS to powerful machine learning and deep learning methods. Classical approaches excel at modelling trends and seasonality under clear assumptions but often falter when faced with non-linear patterns and dynamic data structures. Deep learning models, such as LSTMs and Transformers, address these limitations by learning complex dependencies directly from data without strong prior assumptions. However, they typically lack robust uncertainty quantification, limiting their applicability in risk-sensitive environments.

Bayesian methods have emerged as an alternative that can quantify uncertainty effectively while preserving theoretical interpretability. Yet, traditional Bayesian inference methods, like MCMC sampling, remain computationally intensive and difficult to scale to large, complex datasets. Recent innovations such as ForecastPFN and TabPFN-TS attempt to bridge the gap by leveraging synthetic meta-learning to approximate Bayesian inference in a computationally

efficient manner. Nevertheless, significant challenges remain in fully realising this promise.

Although PFNs demonstrate impressive potential, their application in time series forecasting remains limited and underexplored. Current PFN-based models like ForecastPFN and TabPFN-TS have shown that zero-shot forecasting is feasible; however, these models primarily focus on leveraging structured priors to mimic broad statistical patterns rather than explicitly integrating full Bayesian time series models, such as Bayesian ARIMA, ETS, or dynamic regression structures. Consequently, there is a gap in the literature regarding how rich, domain-specific Bayesian structures can be incorporated into synthetic priors to enhance PFN-based forecasting.

Moreover, existing PFN approaches often use relatively simple synthetic priors and standard feature encodings, such as calendar-based inputs, without fully exploiting hierarchical, multiscale, or autocorrelation-driven structures that are intrinsic to many real-world time series. A prime example is TabPFN-TS, which performs well despite being trained only on generic priors drawn from tabular classification and regression tasks. While impressive, this highlights a key limitation: the lack of targeted exploration into priors specifically structured for time series forecasting. This restricts the generalisation capacity of current models, particularly for datasets exhibiting complex trends, irregular seasonality, or evolving volatility. Addressing this gap—by incorporating priors derived from Bayesian models that are naturally suited to simulate these structures—could significantly advance zero-shot forecasting reliability across broader application domains.

This literature review identifies a compelling need for developing Prior-Data Fitted Networks that integrate richer Bayesian structures specifically designed for time series forecasting. The proposed research will explore how synthetic priors based on Bayesian models can be used to pre-train PFNs, aiming to capture autocorrelation, trend, seasonality, and uncertainty characteristics more faithfully. By constructing priors that more closely reflect domain-specific dynamics, this approach intends to improve both the accuracy and robustness of zero-shot forecasting models.

Additionally, this project will investigate how modifications in feature encoding, such as incorporating autocorrelation statistics, volatility indices, or regime-switching signals, can further enhance PFN performance. Through systematic benchmarking against traditional deep learning and Bayesian forecasting models, this research seeks to demonstrate that Bayesian-enhanced PFNs can offer a scalable, interpretable, and computationally efficient alternative for time series forecasting under uncertainty. By bridging this gap, the project aims to contribute both methodological advances and practical tools for real-world forecasting applications across finance, energy, and logistics sectors.

4 Research Project Plan

4.1 Objective

The primary objective of this research is to develop and evaluate Prior-Data Fitted Networks (PFNs) that incorporate Bayesian time series structures, specifically derived from the Local and Global Trend (LGT) model to enhance zero-shot forecasting performance. The project aims to simulate synthetic training tasks from the posterior predictive distribution of the Bayesian LGT model, embedding realistic temporal dynamics such as nonlinear trends, seasonality, and heteroscedasticity into the PFN meta-learning process. This approach directly addresses the current gap in PFN training, where models are often built on simple, generic priors that fail to capture the structural richness of real-world time series. The study will benchmark the resulting Bayesian-enhanced PFNs against established forecasting methods, including TabPFN-TS, DeepAR, and classical Bayesian models, evaluating improvements in predictive accuracy, uncertainty calibration, and computational efficiency. Ultimately, the project aims to demonstrate that LGT-informed PFNs offer a scalable, interpretable, and data-efficient framework for forecasting across diverse application domains.

4.2 Methodology

4.2.1 Model Selection: Bayesian Time Series Models

To address the gap identified in the literature, this project will construct synthetic priors using the Bayesian Local and Global Trend (LGT) model, which has demonstrated strong performance in forecasting tasks involving nonlinear growth, heteroscedasticity, and seasonal structure. Unlike traditional ARIMA or ETS models, LGT offers a fully Bayesian formulation that captures both local temporal patterns (e.g., short-term deviations) and overarching global dynamics (e.g., long-term trends), making it ideal for simulating time series with realistic uncertainty and structural diversity.

The LGT model generates posterior predictive distributions through methods such as MCMC or Gibbs sampling, producing complete probabilistic forecasts. These forecasts can be used to generate synthetic training datasets that encode domain-specific features—such as varying trend shapes, seasonal amplitudes, and noise volatility—directly into the PFN training process. This approach offers a compelling match for PFNs, which rely on realistic task distributions to learn generalisable forecasting strategies.

4.2.2 Synthetic Prior Construction

Synthetic training datasets will be generated by simulating from the posterior predictive distribution of the Bayesian LGT model. LGT provides a full generative mechanism by producing probabilistic future trajectories based on latent local and global components, allowing the creation of diverse training tasks that mirror the complexity of real-world time series.

For each synthetic dataset, trend parameters, seasonal strengths, local innovation variances, and scale mixture components will be sampled from prior distributions reflecting real-world temporal properties. The LGT framework's ability to encode seasonality, nonlinear growth, and regime-switching behaviours ensures that generated data captures both typical and rare patterns.

Residuals will follow a Student's t-distribution to simulate heavy-tailed volatility, and simulations will vary in horizon length, periodicity (daily, weekly, monthly), and structural characteristics (e.g., damped or additive trends). This method enables the production of thousands of richly structured forecasting tasks, serving as ideal inputs for PFN meta-learning. Compared to earlier PFNs trained on generic or Gaussian noise-based priors, this approach embeds detailed time-series-relevant inductive bias directly into the training pipeline.

4.2.3 Prior-Fitted Network Training

This project adopts the same underlying Transformer-based model and meta-learning framework as TabPFN [20], but repurposes it for a fundamentally different task: zero-shot time series forecasting. While TabPFN was trained on synthetic tabular classification data, our approach trains the model on forecasting tasks generated from structured Bayesian priors—specifically the LGT model—making the training data and objective function entirely regression-based. This shift allows the PFN to learn inductive biases suited to temporal prediction, rather than classification.

During meta-training, the PFN will be exposed to thousands of synthetic time series generated from the LGT model. Each dataset will contain partial observations; the model will use a context-target splitting strategy, masking portions of the data and learning to predict the missing values. These masked forecasts represent regression targets rather than class labels, directly approximating the posterior predictive distribution $p(y \mid x, D)$.

Key input features will include time indices, calendar-based encodings (e.g., day-of-week, month), and optionally lagged observations. The model will be trained using a negative log-likelihood loss over the predictive targets, encouraging accurate point forecasts with calibrated uncertainty. Training will be

performed using mini-batch gradient descent with adaptive learning rates.

Once trained, the PFN will operate in a true zero-shot regime. No fine-tuning will be performed at test time, and hyperparameters will remain fixed during evaluation to ensure comparability with traditional forecasting methods.

4.2.4 Forecasting Benchmarks and Evaluation

To validate the effectiveness of the proposed Bayesian-enhanced PFNs, benchmarking will be conducted against several established models. These will include TabPFN-TS [18], a tabular Prior-Fitted Network adapted for time series prediction; DeepAR [58], a probabilistic LSTM-based forecasting model; and Bayesian statistical models including Local and Global Trend (LGT) and Local-Seasonal-Global Trend (LSGT) models [14], [43], which generalise classical ETS to account for complex growth and seasonality patterns.

Classical Bayesian models, namely Bayesian ARIMA and Bayesian ETS fitted directly to real-world data, will also be included to isolate the value added by the meta-learning approach. Evaluation metrics will consist of:

- Point Forecast Accuracy: measured using sMAPE (Symmetric Mean Absolute Percentage Error) and MASE (Mean Absolute Scaled Error).

- Probabilistic Forecast Quality: assessed through MSIS at 90% and 98% confidence levels (Mean Scaled Interval Score) and CRPS (Continuous Ranked Probability Score).

- Computational Efficiency: evaluated by the average runtime per series, measured in seconds.

This comprehensive evaluation setup ensures that both predictive accuracy and uncertainty calibration are rigorously assessed, alongside practical computational feasibility. Importantly, measuring runtime per series allows us to evaluate a key trade-off: full Bayesian models, while offering robust uncertainty estimation, often suffer from high computational costs due to iterative inference techniques. In contrast, PFN-based approaches approximate posterior predictive distributions in a single forward pass, offering substantial gains in efficiency. By explicitly assessing timing alongside accuracy, the evaluation aims to determine whether the proposed method can offer a viable and scalable alternative to traditional Bayesian inference for time-sensitive forecasting tasks.

4.2.5 Forecasting Tasks

The models will be evaluated across a range of forecasting tasks designed to represent practical real-world scenarios. These will include univariate short-term forecasting, using daily and weekly subsets from the M4 competition dataset, which provides diverse series from various domains including finance, industry, and macroeconomics.

Hierarchical forecasting tasks will involve grouped time series with aggregation constraints, such as sales data aggregated across product categories and geographical regions. Such tasks require models to produce forecasts that remain consistent across different levels of aggregation.

Sparse data forecasting scenarios will feature irregular sampling intervals or missing observations, common in domains such as healthcare monitoring or environmental sensing. Additionally, forecasting problems with limited historical data will be included to assess model performance under data scarcity.

By testing across these diverse settings, the project aims to demonstrate that Bayesian-enhanced PFNs offer flexible and reliable forecasting capabilities even under challenging data conditions, potentially advancing the state-of-the-art in zero-shot time series forecasting.

4.3 Data Sources

The datasets used in this research will be sourced from the Monash Time Series Forecasting Repository [63], an open-access platform offering high-quality time series datasets widely used in forecasting research. In particular, the M4 competition dataset will be utilised, providing a broad collection of real-world time series across domains such as finance, demographics, industry, and tourism. Additionally, the Tourism dataset and the CarParts dataset will be employed to evaluate model performance on domain-specific and sparse time series forecasting tasks. The M4 dataset will enable testing of univariate and hierarchical forecasting capabilities, while the Tourism and CarParts datasets will allow exploration of model robustness in datasets exhibiting seasonal and irregular patterns.

Synthetic datasets generated during the prior construction phase will also serve as training data for Prior-Fitted Networks (PFNs). These datasets will be specifically designed to reflect realistic temporal patterns, including autocorrelation, seasonality, and volatility. All datasets used are publicly available and free from personal identifiers, ensuring compliance with ethical standards and data privacy regulations.

4.4 Timeline and Milestones

The proposed research will be organised into a sequence of structured milestones, each corresponding to a key phase in the project's development. The initial phase will focus on an in-depth review of existing literature, followed by the formulation of synthetic priors grounded in established Bayesian time series models. Once the synthetic priors are defined, the next milestone will involve training Prior-Fitted Networks using datasets generated from the Bayesian Local and Global Trend (LGT) model.

Subsequent steps will include the implementation of benchmarking protocols, where the trained PFN will be compared against several baseline models, including TabPFN-TS, DeepAR, and Bayesian LGT/LSGT variants. This will be followed by a comprehensive evaluation phase to assess predictive accuracy, uncertainty calibration, and computational efficiency using standard forecasting metrics. The final milestone will involve synthesising and analysing results, interpreting findings in relation to the original research aims, and compiling the dissertation for submission. The entire plan is structured to ensure technical feasibility, with each stage building on the prior to enable a coherent, iterative research workflow.

4.5 Ethics

This project involves no human subjects, sensitive personal information, or proprietary datasets, and therefore presents minimal ethical risk. All real-world datasets used, including the M4, Tourism, and CarParts datasets from Monash Time Series Forecasting Repository, are publicly available and anonymised. Synthetic data generated during prior construction will not contain any real-world identifiers. Throughout the project, good research practices will be maintained to ensure transparency, reproducibility, and compliance with academic standards regarding data handling and model evaluation.

5 Conclusion

This project proposes a novel framework that combines Prior-Data Fitted Networks with structured Bayesian time series priors, specifically derived from the Local and Global Trend (LGT) model. By embedding realistic temporal dynamics into the PFN meta-learning process, the approach aims to overcome the limitations of generic priors used in current PFN models. Through rigorous benchmarking against existing deep learning and Bayesian methods, this study will evaluate gains in predictive accuracy, uncertainty quantification, and computational efficiency. The research contributes both a methodological advance

and a practical tool for scalable, interpretable, and zero-shot forecasting across diverse real-world applications.

6 Reference List

[1] G. Mahalakshmi, S. Sridevi, and S. Rajaram, "A survey on forecasting of time series data," in *2016 International Conference on Computing Technologies and Intelligent Data Engineering (ICCTIDE'16)*, Kovilpatti, India: IEEE, Jan. 2016, pp. 1–8. doi: https://doi.org/10.1109/ICCTIDE.2016.7725358.

[2] R. J. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice*, 2nd ed. OTexts: Melbourne, Australia, 2018. Accessed: Mar. 24, 2025. [Online]. Available: OTexts.com/fpp2

[3] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "M5 accuracy competition: Results, findings, and conclusions," *International Journal of Forecasting*, vol. 38, no. 4, pp. 1346–1364, 2022, doi: https://doi.org/10.1016/j.ijforecast.2021.11.013.

[4] S. J. Taylor and B. L. and, "Forecasting at Scale," *The American Statistician*, vol. 72, no. 1, pp. 37–45, 2018, doi: https://doi.org/10.1080/00031305.2017.1380080.

[5] D. C. Montgomery, C. L. Jennings, and M. Kulahci, *Introduction to Time Series Analysis and Forecasting*, 3rd ed. Wiley, 2024.

[6] V. R. Chimmula, "A Data Driven Based Comparison Study of Statistical and Deep Learning Based Time Series Forecasting Methods for Infectious Disease Modeling and Financial Data," PhD Thesis, 2021. [Online]. Available: https://www.proquest.com/dissertations-theses/data-driven-based-comparison-study-statistical/docview/2611626912/se-2

[7] S. Smyl, "A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting," *International Journal of Forecasting*, vol. 36, no. 1, pp. 75–85, 2020, doi: https://doi.org/10.1016/j.ijforecast.2019.03.017.

[8] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: https://doi.org/10.1162/neco.1997.9.8.1735.

[9] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The M4 Competition: 100,000 time series and 61 forecasting methods," *International Journal of Forecasting*, vol. 36, no. 1, pp. 54–74, 2020, doi: https://doi.org/10.1016/j.ijforecast.2019.04.014.

[10] R. Wang, Y. Xin, Y. Zhang, F. Perez-Cruz, and M. Raubal, "Counterfactual Explanations for Deep Learning-Based Traffic Forecasting." 2024. [Online]. Available: https://arxiv.org/abs/2405.00456

[11] Y. Xin, Y. Hong, S. Dirmeier, F. Perez-Cruz, and M. Raubal, "Evaluating the Robustness of Deep Learning Models for Mobility Prediction Through Causal Interventions." ETH Zurich, Zurich, Jun. 2023. doi: https://doi.org/10.3929/ethz-b-000615973.

[12] C. Fernández-Loría, F. Provost, and X. Han, "Explaining Data-Driven Decisions made by AI Systems: The Counterfactual Approach." 2021. [Online]. Available: https://arxiv.org/abs/2001.07417

[13] "Nature of Bayesian Inference," in *Bayesian Inference in Statistical Analysis*, John Wiley & Sons, Ltd, 1992, pp. 1–75. doi: https://doi.org/10.1002/9781118033197.ch1.

[14] M. F. Steel, "Bayesian time series analysis," in *Macroeconometrics and time series analysis*, Springer, 2010, pp. 35–45.

[15] S. Smyl, C. Bergmeir, A. Dokumentov, X. Long, E. Wibowo, and D. Schmidt, "Local and global trend Bayesian exponential smoothing models," *International Journal of Forecasting*, vol. 41, no. 1, pp. 111–127, 2025, doi: https://doi.org/10.1016/j.ijforecast.2024.03.006.

[16] S. Putcha, "Scalable Bayesian Inference using Stochastic Gradient Markov Chain Monte Carlo." Order No. 31824791, Lancaster University (United Kingdom), England, 2024.

[17] S. Muller, N. Hollmann, S. P. Arango, J. Grabocka, and F. Hutter, "Transformers Can Do Bayesian Inference," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=KSugKcbNf9

[18] S. B. Hoo, S. Müller, D. Salinas, and F. Hutter, "The Tabular Foundation Model TabPFN Outperforms Specialized Time Series Forecasting Models Based on Simple Features." 2024. [Online]. Available: https://arxiv.org/abs/2501.02945

[19] S. Dooley, G. S. Khurana, C. Mohapatra, S. V. Naidu, and C. White, "ForecastPFN: Synthetically-Trained Zero-Shot Forecasting," in *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., Curran Associates, Inc., 2023, pp. 2403–2426. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2023/file/0731f0e65559059eb9cd9d6f44ce2dd8-Paper-Conference.pdf

[20] N. Hollmann, S. Müller, K. Eggensperger, and F. Hutter, "TabPFN: A Transformer That Solves Small Tabular Classification Problems in a Second." 2023. [Online]. Available: https://arxiv.org/abs/2207.01848

[21] D. C. Montgomery, C. L. Jennings, and M. Kulahci, *Introduction to Time Series Analysis and Forecasting*, 3rd ed. Wiley, 2009.

[22] G. E. P. Box and G. M. Jenkins, *Time series analysis; forecasting and control*. in Holden-Day series in time series analysis. San Francisco: Holden-Day, 1970.

[23] R. H. Shumway and D. S. Stoffer, "ARIMA Models," in *Time Series Analysis and Its Applications: With R Examples*, Cham: Springer International Publishing, 2017, pp. 75–163. doi: https://doi.org/10.1007/978-3-319-52452-8_3.

[24] J. Kaur, K. S. Parmar, and S. Singh, "Autoregressive models in environmental forecasting time series: a theoretical and application review," *Environmental Science and Pollution Research*, vol. 30, no. 8, pp. 19617–19641, Feb. 2023, doi: 10.1007/s11356-023-25148-9.

[25] A. A. Ariyo, A. O. Adewumi, and C. K. Ayo, "Stock Price Prediction Using the ARIMA Model," in *2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*, 2014, pp. 106–112. doi: https://doi.org/10.1109/UKSim.2014.67.

[26] D. Benvenuto, M. Giovanetti, L. Vassallo, S. Angeletti, and M. Ciccozzi, "Application of the ARIMA model on the COVID-2019 epidemic dataset," *Data in Brief*, vol. 29, p. 105340, 2020, doi: https://doi.org/10.1016/j.dib.2020.105340.

[27] R. J. Hyndman and Y. Khandakar, "Automatic Time Series Forecasting: The forecast Package for R," *Journal of Statistical Software*, vol. 27, no. 3, pp. 1–22, 2008, doi: https://doi.org/10.18637/jss.v027.i03.

[28] J. G. D. Gooijer and K. Kumar, "Some recent developments in non-linear time series modelling, testing, and forecasting," *International Journal of Forecasting*, vol. 8, no. 2, pp. 135–156, 1992, doi: https://doi.org/10.1016/0169-2070(92)90115-P.

[29] J. W. Taylor, "Short-term electricity demand forecasting using double seasonal exponential smoothing," *Journal of the Operational Research Society*, vol. 54, no. 8, pp. 799–805, 2003, doi: https://doi.org/10.1057/palgrave.jors.2601589.

[30] C. Chatfield, "Calculating Interval Forecasts," *Journal of Business & Economic Statistics*, vol. 11, no. 2, pp. 121–135, Apr. 1993, doi: https://doi.org/10.1080/07350015.1993.10509938.

[31] H. S. Lee, M. Her, M. Levine, and G. E. Moore, "Time series analysis of human and bovine brucellosis in South Korea from 2005 to 2010," *Preventive*

Veterinary Medicine, vol. 110, no. 2, pp. 190–197, 2013, doi: https://doi.org/10.1016/j.prevetmed.2012.12.003.

[32] A. J. Wahyudi and F. Febriani, "Time-series forecasting of particulate organic carbon on the Sunda Shelf: Comparative performance of the SARIMA and SARIMAX models," *Regional Studies in Marine Science*, vol. 80, p. 103863, 2024, doi: https://doi.org/10.1016/j.rsma.2024.103863.

[33] S. I. Gass and C. M. Harris, *Encyclopedia of Operations Research and Management Science*, Centennial ed. Dordrecht, The Netherlands: Kluwer, 2000.

[34] R. J. Hyndman, A. B. Koehler, R. D. Snyder, and S. Grose, "A state space framework for automatic forecasting using exponential smoothing methods," *International Journal of Forecasting*, vol. 18, no. 3, pp. 439–454, 2002, doi: https://doi.org/10.1016/S0169-2070(01)00110-8.

[35] R. Hyndman, A. B. Koehler, J. K. Ord, and R. D. Snyder, *Forecasting with exponential smoothing: the state space approach*. Springer Science & Business Media, 2008.

[36] G. Yapar, İ. Yavuz, and H. Taylan Selamlar, "Why and how does exponential smoothing fail? An in depth comparison of ATA-simple and simple exponential smoothing.," *Turkish Journal of Forecasting*, vol. 01, no. 1, pp. 30–39, 2017.

[37] P. J. Harrison and C. F. Stevens, "Bayesian Forecasting," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 38, no. 3, pp. 205–228, Dec. 2018, doi: https://doi.org/10.1111/j.2517-6161.1976.tb01586.x.

[38] M. West and J. Harrison, *Bayesian Forecasting and Dynamic Models*, 1st ed. in Springer Series in Statistics. Springer New York, NY. [Online]. Available: https://doi.org/10.1007/978-1-4757-9365-9

[39] A. E. Gelfand, S. E. Hills, A. Racine-Poon, and A. F. M. S. and, "Illustration of Bayesian Inference in Normal Data Models Using Gibbs Sampling," *Journal of the American Statistical Association*, vol. 85, no. 412, pp. 972–985, 1990, doi: https://doi.org/10.1080/01621459.1990.10474968.

[40] S. Chib and E. G. and, "Understanding the Metropolis-Hastings Algorithm," *The American Statistician*, vol. 49, no. 4, pp. 327–335, 1995, doi: https://doi.org/10.1080/00031305.1995.10476177.

[41] J. Marriott and P. Newbold, "Bayesian Comparison of ARIMA and Stationary ARMA Models," *International Statistical Review / Revue Internationale de Statistique*, vol. 66, no. 3, pp. 323–336, 1998.

[42] C. S. Forbes, R. D. Snyder, and R. G. Shami, "Bayesian exponential smooth-

ing," Monash University, Department of Econometrics and Business Statistics, 2000.

[43] X. Long, D. F. Schmidt, C. Bergmeir, and S. Smyl, "Fast Gibbs sampling for the local and global trend Bayesian exponential smoothing model." 2024. [Online]. Available: https://arxiv.org/abs/2407.00492

[44] D. M. Blei, A. Kucukelbir, and J. D. M. and, "Variational Inference: A Review for Statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, pp. 859–877, 2017, doi: https://doi.org/10.1080/01621459.2017.1285773.

[45] H. Rue, S. Martino, and N. Chopin, "Approximate Bayesian Inference for Latent Gaussian models by using Integrated Nested Laplace Approximations," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 71, no. 2, pp. 319–392, Apr. 2009, doi: https://doi.org/10.1111/j.1467-9868.2008.00700.x.

[46] A. Lapedes and R. Farber, "Nonlinear signal processing using neural networks: Prediction and system modelling," Jun. 1987. [Online]. Available: https://www.osti.gov/biblio/5470451

[47] A. S. Weigend, B. A. Huberman, and D. E. Rumelhart, "PREDICTING THE FUTURE: A CONNECTIONIST APPROACH," *International Journal of Neural Systems*, vol. 01, no. 03, pp. 193–209, 1990, doi: https://doi.org/10.1142/S0129065790000102.

[48] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994, doi: https://doi.org/10.1109/72.279181.

[49] S. Siami-Namini, N. Tavakoli, and A. Siami Namin, "A Comparison of ARIMA and LSTM in Forecasting Time Series," in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2018, pp. 1394–1401. doi: https://doi.org/10.1109/ICMLA.2018.00227.

[50] K. Bandara, C. Bergmeir, and S. Smyl, "Forecasting across time series databases using recurrent neural networks on groups of similar series: A clustering approach," *Expert Systems with Applications*, vol. 140, p. 112896, 2020, doi: https://doi.org/10.1016/j.eswa.2019.112896.

[51] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *European Journal of Operational Research*, vol. 270, no. 2, pp. 654–669, 2018, doi: https://doi.org/10.1016/j.ejor.2017.11.054.

[52] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the Proper-

ties of Neural Machine Translation: Encoder-Decoder Approaches." 2014. [Online]. Available: https://arxiv.org/abs/1409.1259

[53] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling." 2014. [Online]. Available: https://arxiv.org/abs/1412.3555

[54] A. Vaswani et al., "Attention Is All You Need." 2023. [Online]. Available: https://arxiv.org/abs/1706.03762

[55] B. Lim, S. Ö. Arık, N. Loeff, and T. Pfister, "Temporal Fusion Transformers for interpretable multi-horizon time series forecasting," *International Journal of Forecasting*, vol. 37, no. 4, pp. 1748–1764, 2021, doi: https://doi.org/10.1016/j.ijforecast.2021.03.012.

[56] H. Zhou, "Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting", *AAAI*, vol. 35, no. 12, pp. 11106-11115, May 2021.

[57] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, "DeepAR: Probabilistic forecasting with autoregressive recurrent networks," *International Journal of Forecasting*, vol. 36, no. 3, pp. 1181–1191, 2020, doi: https://doi.org/10.1016/j.ijforecast.2019.07.001.

[58] S. S. Rangapuram, M. W. Seeger, J. Gasthaus, L. Stella, Y. Wang, and T. Januschowski, "Deep State Space Models for Time Series Forecasting," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., Curran Associates, Inc., 2018. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2018/file/5cf68969fb67aa6082363a6d4e6468e2-Paper.pdf

[59] L. V. Jospin, H. Laga, F. Boussaid, W. Buntine, and M. Bennamoun, "Hands-On Bayesian Neural Networks—A Tutorial for Deep Learning Users," *IEEE Computational Intelligence Magazine*, vol. 17, no. 2, pp. 29–48, 2022, doi: https://doi.org/10.1109/MCI.2022.3155327.

[60] S. Fort, H. Hu, and B. Lakshminarayanan, "Deep Ensembles: A Loss Landscape Perspective." 2020. [Online]. Available: https://arxiv.org/abs/1912.02757

[61] B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, "N-BEATS: Neural basis expansion analysis for interpretable time series forecasting." 2020. [Online]. Available: https://arxiv.org/abs/1905.10437

[62] T. Nagler, "Statistical foundations of prior-data fitted networks," in *International Conference on Machine Learning*, PMLR, 2023, pp. 25660–25676. [Online]. Available: https://arxiv.org/abs/2305.11097

R. Godahewa, C. Bergmeir, G. I. Webb, R. J. Hyndman, and P. Montero-Manso, "Monash Time Series Forecasting Archive," in *Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021. [Online]. Available: `https://forecastingdata.org/`

Part II

Research Paper

Combining Full Bayesian Inference with Prior-fitted Networks for Time Series Forecasting

Abstract

Prior-Fitted Networks (PFNs) offer a novel approach to time series forecasting by training neural networks on synthetically generated data from statistical models, enabling zero-shot predictions without fine-tuning. This study investigates whether the Local and Global Trend (LGT) model, a Bayesian exponential smoothing method, can serve as an effective prior for PFN training. LGT models are fitted to M3 monthly competition series, and their learned parameter distributions are used to generate synthetic time series for training a ForecastPFN transformer architecture. Evaluation across five datasets reveals mixed performance: the model generally underperforms traditional automatic methods like AutoETS and AutoARIMA, but achieves competitive results on specific datasets. These findings demonstrate that LGT-generated synthetic data can capture meaningful forecasting patterns, whilst highlighting the importance of synthetic data diversity and prior-architecture alignment. The research establishes a foundation for combining Bayesian statistical models with neural forecasting approaches in the zero-shot domain.

1 Introduction

Time series forecasting is the process of predicting future values based on historical observations and appears across domains from electricity demand to stock prices [1]. Traditional statistical methods like ARIMA [2] and Exponential Smoothing [3] remain widely used for their interpretability and effectiveness, with modern implementations such as AutoARIMA [4] and AutoETS

[5] enabling automatic model selection. Recent deep learning methods have gained attention for learning patterns across multiple series [6], yet many struggle to consistently outperform traditional methods, especially on shorter series or limited data [7]. More troubling, recent machine learning forecasting literature exhibits serious methodological shortcomings: widespread misuse of scale-dependent metrics on normalised data that obscures true performance [8], [9], inadequate validation procedures violating established standards, and poor reproducibility. Prior-Fitted Networks (PFNs) [10] offer a different paradigm: training on synthetic data from prior distributions rather than specific datasets, then performing approximate Bayesian inference at test time in a single forward pass for zero-shot generalisation. Existing PFNs use varied priors. TabPFN [11] trains on synthetic tabular data unrelated to time series, while ForecastPFN [12] trains on Gaussian Process-based synthetic time series, claiming competitive performance. However, as detailed in Section 3.2.3 below, we identify multiple critical flaws in ForecastPFN: training divergence, improper input scaling causing training to fail due to numerical instability, evaluation on normalised data violating best practices, evaluation on single time series (rather than complete datasets per standard practice), and unusually "flat" forecasts when applied to standard forecasting benchmark datasets. These problems fundamentally undermine reported results and raise questions about rigour in PFN forecasting research more broadly.

This work makes two key contributions. First, we investigate whether training on synthetic data from a time series-specific Bayesian model improves performance for PFN forecasting models. The Long-term, Global Trend (LGT) model [13], [14] is used as the prior. Unlike Gaussian Processes or unrelated tabular data as used in existing approaches, LGT explicitly models fundamental time series components: trends, seasonality, level shifts, robust errors through a principled Bayesian framework. Crucially, LGT enables generating synthetic data from distributions conditioned on real data by fitting LGT to actual time series, obtaining parameter distributions capturing realistic characteristics, then sampling from these distributions to generate training data reflecting real-world patterns rather than purely theoretical assumptions. In this work, training is conducted on 142,800 synthetic series generated from LGT fitted to M3 monthly data [15] with evaluation against AutoETS, AutoARIMA, and Seasonal Naive using proper scale-independent metrics, MASE [16] and sMAPE [17] on original-scale data. Second, multiple serious methodological issues affecting ForecastPFN's are identified and systematically addressed and resolved. This reveals that the results claimed in the original ForecastPFN paper likely do not reflect genuine capability, highlighting the importance of methodological rigour in forecasting research. In identifying and addressing these problems, this work lays the foundation for further research applying PFN models to forecasting. The rest of this paper is structured as follows: Section 2 discusses relevant background material, Section 3 sets out our methodology, Section 4 outlines the experimental setup and presents the corresponding results and discussion, and Section 5 set our conclusions.

2 Background

2.1 Time Series Forecasting

Time series forecasting predicts future values of a sequence based on historical observations ordered in time. Formally, given observations $y_1, y_2, ..., y_t$, the task is to predict $y_{t+1}, y_{t+2}, ..., y_{t+h}$ where h is the forecast horizon [18]. Time series exhibit temporal dependence where observations are correlated across time, systematic patterns including trend and seasonality, and irregular noise components [19].

Statistical methods remain widely used in practice. Autoregressive Integrated Moving Average (ARIMA) models [2] capture temporal dependencies through autoregressive and moving average components. Exponential Smoothing State Space Models (ETS) [20] emphasise recent observations while exponentially decaying weights of older data. These methods offer well-developed theory, interpretable parameters, and uncertainty quantification through prediction intervals. Machine learning approaches, particularly deep learning methods such as Recurrent Neural Networks (RNNs), Long Short-Term Memory networks (LSTMs) [21], and Transformers [22], can potentially learn complex nonlinear patterns and leverage information from multiple related time series during training. However, they typically require large amounts of training data and can be challenging to tune and interpret [23].

A key challenge is generalisation: models must perform well on future unseen values that may exhibit patterns not fully represented in historical data. This is particularly difficult for short time series with limited observations (typical of, e.g., time series consisting of monthly or yearly data). Additionally, forecasting performance often degrades as the forecast horizon increases due to accumulating uncertainty [24]. These challenges motivate methods that can effectively learn from diverse time series and adapt to new forecasting tasks with minimal data requirements.

2.2 Prior-fitted Networks

Prior-Fitted Networks (PFNs) represent a shift in machine learning by inverting the traditional training approach. Instead of training a model on observed data to make predictions on new instances, PFNs are trained on synthetic data sampled from a prior distribution and learn to perform Bayesian inference at test time [10], enabling predictions on entirely new datasets without gradient-based optimisation or fine-tuning.

2.2.1 Theoretical Foundation

The fundamental idea behind PFNs is rooted in Bayesian inference. In supervised learning, the Bayesian approach views the data-generating process through a prior distribution $p(t)$ over a latent variable t which is the task. Given a training dataset $D = \{(x_i, y_i)\}_{i=1}^{n}$ and a new input x, the goal is to compute the posterior predictive distribution (PPD):

$$p(y \mid x, D) = \int_t p(y \mid x, t)\, p(t \mid D)\, dt$$

where $p(t \mid D)$ is the posterior distribution over tasks given observed data [10]. Computing this integral is typically intractable, requiring approximations like MCMC [25] or variational inference [26].

PFNs approximate this Bayesian inference procedure by training a parameterised neural network q_θ to directly predict the PPD. The training objective, called Prior-Data Negative Log-Likelihood (Prior-Data NLL), is:

$$\ell_\theta = \mathbb{E}_{D \cup \{x,y\} \sim p(D)}[-\log q_\theta(y \mid x, D)]$$

where $D \cup \{x, y\}$ denotes a dataset sampled from $p(D)$ [10]. Minimising ℓ_θ is equivalent to minimising the expected KL-Divergence between the true PPD and its approximation q_θ [27], yielding an approximation of the PPD.

A Simplified View: In practical terms, PFNs invert the traditional machine learning paradigm. Instead of training on individual examples (x_i, y_i) sampled from a single dataset, a PFN trains on entire datasets D sampled from the prior distribution. At inference time, the PFN takes the entire available "training" dataset as input and makes predictions conditioned on this complete context. This enables in-context learning: the network adapts its predictions based on provided data without parameter updates, similar to how large language models solve new tasks from examples in prompts.

Training Algorithm

The foundation PFN algorithm can be defined as per Algorithm 1. The only requirement for PFNs is the ability to sample from the prior distribution $p(D)$. Unlike MCMC or variational inference, PFNs only need to generate synthetic datasets from the prior [10].

Algorithm 1 Training a Prior-Fitted Network [10]

Input: A prior distribution $p(\mathcal{D})$ over datasets, total number of samples K.
Output: A neural model q_θ approximating the PPD.

Initialise network parameters θ for q_θ;

1: **for** $j = 1$ to K **do**
2: Draw dataset D and batch $\{(x_i, y_i)\}_{i=1}^{m}$ from $p(\mathcal{D})$;
3: Compute loss $\bar{\ell}_\theta = \sum_{i=1}^{m} -\log q_\theta(y_i \mid x_i, D)$;
4: Update θ via stochastic gradient descent using $\nabla_\theta \bar{\ell}_\theta$;
5: **end for**

2.2.2 Properties and Advantages

PFNs offer several distinctive advantages over traditional machine learning approaches, at least in principle:

Zero-shot generalisation: Once trained, PFNs can immediately make predictions on new datasets without additional training, fine-tuning, or hyperparameter selection, eliminating computational costs and cross-validation needs for each new task [10].

Uncertainty quantification: By approximating Bayesian inference, PFNs naturally provide probabilistic predictions and uncertainty estimates valuable for decision-making [10].

Sample efficiency: PFNs can make reasonable predictions from small datasets by leveraging knowledge encoded in the prior rather than relying solely on observed data [10].

Computational efficiency at inference: While training requires significant resources, inference is extremely fast, typically a single forward pass through the network [12].

In-context learning: PFNs adapt predictions based on provided datasets without explicit parameter updates, similar to large language models [10].

However, performance depends critically on prior distribution choice. The PFN's approximation quality is bounded by how well the prior captures real-world data-generating processes [10], [27].

2.2.3 Different Types of Priors and Applications

The choice of prior distribution fundamentally determines what patterns a PFN recognises and how it generalises. Different variants have explored various prior specifications, for example:

TabPFN [11] addresses small-scale tabular classification by training on synthetic datasets from a structured prior combining feature distributions, label-generating functions (linear models, decision trees, neural networks), and noise models. A single trained model outperformed XGBoost and AutoML systems on datasets with up to 1,000 samples without hyperparameter tuning. **TabPFN-TS** [28] showed the same pre-trained model could be applied to time series forecasting by transforming temporal data into tabular format.

ForecastPFN [12] extended PFNs to time series using synthetic data from Gaussian Processes. The prior models time series as $y_t = trend(t) \cdot seasonal(t) \cdot z(t)$, combining trend, seasonal, and Weibull-based noise components. ForecastPFN claimed competitive performance against established baselines with 100x faster inference compared to other transformer models.

LGT offers a principled Bayesian prior for time series, explicitly modelling trends, seasonality, and robust errors [13], [14]. Crucially, by fitting LGT to real series, posterior parameter distributions can be sampled to generate synthetic data reflecting real-world dynamics. Unlike Gaussian Processes which use fixed distributions or TabPFN which uses unrelated data, this conditions synthetic generation on actual forecasting patterns, enabling PFNs to learn from realistic data characteristics.

2.3 Full Bayesian Inference & LGT model

2.3.1 Introduction to Full Bayesian Inference in Time Series

Full Bayesian inference treats model parameters as random variables with probability distributions rather than fixed values [29]. This approach quantifies uncertainty through posterior distributions, combining prior beliefs with data evidence to produce probabilistic forecasts [30]. For LGT, the Bayesian framework enables handling heavy-tailed errors through Student-t distributions, heteroscedastic variance structures, and complex non-linear trends [13].

2.3.2 The LGT Model Structure

The Local and Global Trend (LGT) model extends classical exponential smoothing to capture trends growing faster than linear but slower than exponential [13]. This research employs the implementation by Long *et al.* [14], offering efficient simulation whilst preserving essential modelling principles.

LGT models each observation using a Student t-distribution, providing robustness to outliers through heavier tails controlled by parameter ν [31]. The forecasting equation combines a smoothed level, global trend term, and optional local trend. The trend generalisation takes the form γl_t^{ρ} where l is the smoothed level, γ is the global trend coefficient, and ρ determines growth rate: near zero for linear growth, approaching one for exponential growth [14]. Level and local trend evolve through exponential smoothing with parameters α and β estimated within the Bayesian framework. For seasonal data, multiplicative seasonal factors are smoothed on a logarithmic scale, and heteroscedastic errors are accommodated through a scale parameter varying with series level [14].

2.3.3 LGT's Suitability as a Prior for PFN

LGT's Bayesian formulation provides a natural mechanism for generating diverse synthetic training data by sampling parameter combinations from posterior distributions. The model's interpretable parameters, including ρ for trend shape, ν for tail heaviness, τ for error scaling, can be systematically varied to produce series with different characteristics [13]. The `rlgt` package implementation [14] enables efficient large-scale simulation through Gibbs-based sampling algorithms, making it computationally feasible to generate the thousands of synthetic series required for PFN training.

2.4 Time Series Forecasting Evaluation

2.4.1 The Problem with Scale-Dependent Metrics

A fundamental issue exists in the machine learning literature on time series forecasting: the widespread use of scale-dependent error metrics on inappropriately scaled data. Metrics such as Mean Squared Error (MSE) and Mean Absolute Error (MAE) depend on the scale of the data being forecast [8]. An MAE of 10 is excellent when forecasting values in the thousands but disastrous when forecasting values between 0 and 100. This makes these metrics unsuitable for comparing forecast accuracy across multiple time series with different scales.

The problem intensifies when researchers evaluate models on standardised or normalised data: a common practice in recent deep learning forecasting papers. For instance, ForecastPFN [12], TimePFN [32], Informer [33], and FEDformer [34] all report MSE or MAE as primary evaluation metrics, often on normalised data. While z-score normalisation may aid neural network training, evaluating on the transformed scale obscures true forecast performance. A model achieving MSE of 0.5 on normalised data tells us nothing about its performance on the original scale that matters for decision-making. This approach also assumes future values will have the same mean and variance as training data, which is an assumption frequently violated in non-stationary environments.

2.4.2 Scale-Independent Alternatives

The Mean Absolute Scaled Error (MASE) addresses these limitations by scaling forecast errors relative to the in-sample MAE of a naive forecast [8]. It is defined as:

$$\text{MASE} = \frac{1}{n}\sum_{t=1}^{n}\frac{|y_t - \hat{y}_t|}{\frac{1}{T-m}\sum_{i=m+1}^{T}|y_t - y_{t-m}|}$$

where the denominator represents the in-sample MAE of the one-step naive forecast. A MASE below one indicates performance better than the naive benchmark on the test data; values above one indicate worse performance. Lower MASE values represent more accurate forecasts. Crucially, MASE is independent of data scale and well-defined even when series contain zero values: a situation that breaks percentage-based metrics.

The Symmetric Mean Absolute Percentage Error (sMAPE) provides an alternative scale-independent measure:

$$\text{sMAPE} = \frac{100\%}{n}\sum_{t=1}^{n}\frac{|y_t - \hat{y}_t|}{(|y_t| + |\hat{y}_t|)/2}$$

Despite its name suggesting symmetry, sMAPE is not truly symmetric: for the same actual value, the metric penalises under-forecasts more heavily than over-forecasts [8]. Additionally, when actual values approach zero, the forecast is also likely near zero, leading to division by numbers close to zero and potentially unstable or undefined values. Despite these limitations, sMAPE remains a valuable scale-independent metric and is widely used in forecasting competitions and literature [35]. We will use MASE and sMAPE in this paper to provide comprehensive evaluation and facilitate comparison with existing forecasting research.

3 Methodology

The methodology consists of three main components: (1) generating synthetic training data by fitting LGT to M3 monthly series and sampling from learned parameter distributions, (2) training the ForecastPFN architecture on this synthetic data (having identified and corrected multiple serious flaws in the original ForecastPFN implementation), and (3) using the ForecastPFN model trained on the simulated data to make predictions and evaluating the results.

3.1 Using LGT as a Prior for Synthetic Data Generation

3.1.1 From Forecasting Model to Synthetic Data Generator

In this work, we adapted LGT from a forecasting model into a synthetic data generator for training Prior-Fitted Networks. Specifically, we fit LGT to each M3 monthly training series (further details in relation to the training data are provided in Section 3.3.1 below), extracted the posterior parameter distributions from these fits, and drew samples from these distributions to generate diverse synthetic time series. This approach produces training data conditioned on real-world forecasting patterns rather than purely theoretical priors.

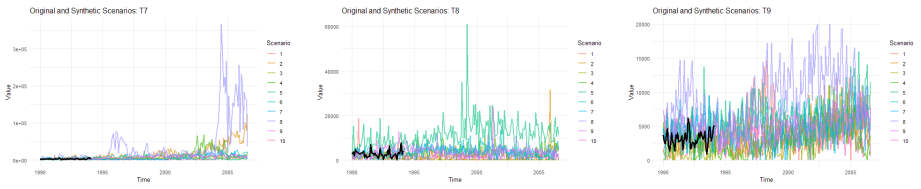
Traditional forecasting methods optimised via maximum likelihood produce point estimates of parameters. While suitable for prediction, these offer limited simulation capability. Generating synthetic data requires either fixing parameters at point estimates, which produces unrealistic uniformity, or introducing random adjustments lacking principled justification. LGT's Bayesian approach fundamentally differs by producing full posterior distributions over all model parameters [14].

These posterior distributions represent uncertainty about true parameter values given observed data. For simulation, this uncertainty becomes a strength: sampling parameter combinations from posteriors generates synthetic series exhibiting realistic variability. Each sampled parameter set represents a plausible model configuration, producing training data with the heterogeneity necessary for robust neural network learning. The `rlgt` package implementation makes large-scale simulation computationally feasible through efficient Gibbs-based inference and fast forward simulation [14].

3.1.2 Simulation Process

Synthetic training data were generated in two stages: fitting LGT models and sampling synthetic series from the fitted models. We fit LGT individually

to each of the 1,428 M3 monthly training series using the `rlgt` package [14], with test portions excluded to prevent data leakage. Each fitted model captures the specific characteristics of its corresponding series through posterior parameter distributions over trend shape, seasonality, error variance, and other LGT components. For each fitted LGT model, 100 distinct synthetic scenarios were generated, each consisting of a 200-point time series. Rather than forecasting beyond the original series, simulation begins from the start date of each original series with varied initial conditions. The initial level is sampled from observed values with randomness added from a Student-t distribution with 3 degrees of freedom [31], ensuring some scenarios begin with substantially different starting points that reflect real-world uncertainty and occasional extreme deviations. The initial trend is randomly selected from observed differences in the series, capturing the range of growth rates in the original data. For seasonal component, initial seasonal factors are derived from the original series. Within each scenario, 100 candidate paths are generated using LGT's forward simulation mechanism, and one is randomly selected. This two-stage sampling across scenarios and within scenarios ensures the synthetic training set exhibits realistic diversity while remaining anchored to statistical properties learned from M3 data. With 1,428 M3 monthly series generating 100 synthetic series each, the process yields 142,800 synthetic time series for PFN training, each with length 200. Sample series can be found below (Figure 1) and in Appendix A.1.



(a) T7 (b) T8 (c) T9

Figure 1: Examples of generated sample series. Only 10 samples per plot are displayed for clearer visualisation. The black line is the original series used for fitting.

3.1.3 Using M3 Monthly Data for Learning Parameter Distributions

We use the M3 monthly data to parameterise LGT and serve as the foundation for simulated data used in PFN training. The M3 competition dataset contains 1,428 monthly series spanning industry, finance, demographics, and macroeconomics [15]. By fitting LGT to this diverse collection, parameter distributions are extracted that encode varied real-world forecasting patterns rather than characteristics of a single narrow domain. M3 monthly series offer practical advantages for LGT fitting: sufficient observations for reliable parameter estimation while maintaining computational efficiency. Predetermined train/test

splits prevent data leakage. LGT is fitted only to training portions, parameter distributions are extracted, synthetic training data are generated, and PFN performance is evaluated on withheld test sets. This ensures that the PFN never accesses information from test series during training, maintaining evaluation integrity. The resulting synthetic dataset combines structure learned from real forecasting tasks with controlled randomisation, producing training data diverse enough to prevent overfitting yet realistic enough to generalise to actual applications.

3.2 Neural Network Architecture

3.2.1 ForecastPFN Overview

The ForecastPFN model [12] serves as the basis for the experiments in training a prior-fitted network using data simulated from LGT. ForecastPFN was selected as it represents one of the prominent recent methods applying PFN models to forecasting, demonstrating competitive results on multiple standard benchmarks. Its transformer-based architecture and training procedure are well-suited for adaptation to LGT-simulated data: the model is designed to learn from diverse synthetic training series and perform zero-shot inference on new datasets, aligning directly with the objective of training on LGT simulations and evaluating on real benchmarks without fine-tuning. Figure 2 illustrates the ForecastPFN training process: the model trains once on diverse simulated series, then applies zero-shot to new, unseen real-world series. The architecture processes historical time series data along with temporal features through a transformer encoder, then predicts future values conditioned on target timestamps.

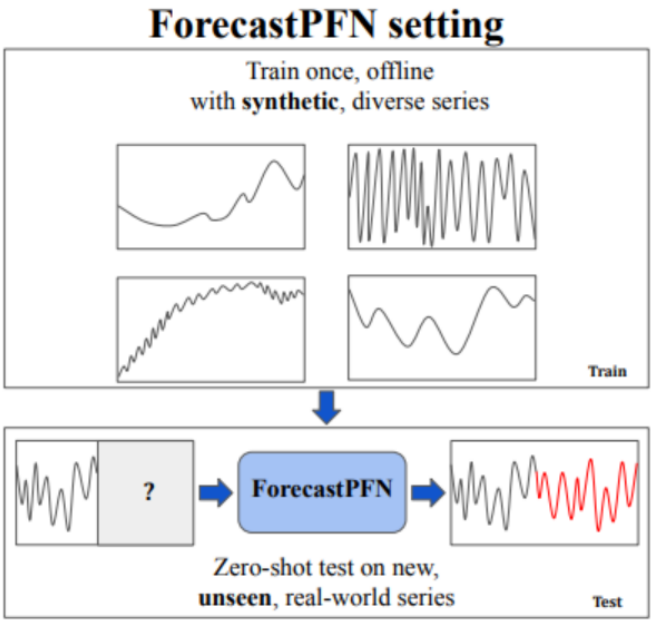


Figure 2: ForecastPFN training paradigm: train once on simulated data, apply zero-shot to real series

The original approach includes a noise component in synthetic data generation, multiplying history values with noise factors during training. Since LGT-simulated data already incorporates stochasticity through the Bayesian parameter sampling and forward simulation process, we set all noise multipliers to 1, effectively using the LGT simulations directly without additional noise injection. This adjustment ensures that the model learns from the realistic variability inherent in LGT's statistical framework rather than from artificial multiplicative noise.

3.2.2 Architecture Components

The ForecastPFN architecture is used largely as implemented in the original paper, with critical corrections to training procedures and evaluation protocols. The base architecture remains unchanged. It consists of three main components: input embedding, transformer encoding, and output prediction, each described here for completeness.

The input embedding layer processes both historical time series values and their associated temporal features. We scale the values using max scaling:

scaled = history/(max(history) + ϵ), where $\epsilon = 10^{-4}$ prevents division by zero. This scaling enables the model to handle series of different magnitudes without requiring external normalisation. Note that the original ForecastPFN paper employed a custom "robust scaler"; however, this implementation was found to cause gradient explosion during training (even on the original training data claimed to be used in the ForecastPFN paper). The simpler max scaling approach used here resolves this training instability while maintaining scale-invariance.

Temporal information is encoded through position embeddings that capture multiple time scales. Separate positional encodings are constructed for year (10 periods, 4 frequencies), month (12 periods, 4 frequencies), day (31 periods, 6 frequencies), and day of week (7 periods, 4 frequencies) using sine and cosine functions: $\sin(\pi/P \cdot 2^j \cdot (i-1))$ and $\cos(\pi/P \cdot 2^j \cdot (i-1))$, where P is the period length, j indexes the frequency, and i is the position. These embeddings are concatenated to form a 36-dimensional temporal representation for each time point. This approach is broadly similar to the index-like independent variables used in other PFN forecasting architectures such as TabPFN-TS [28], providing the model with explicit temporal context.

The historical values and temporal features are processed through separate dense layers before combination. The scaled history passes through two parallel dense layers: one that ignores positional information and another that incorporates it by adding the positional embeddings. Both outputs are concatenated, yielding a 72-dimensional embedding for each historical time point. Similarly, target timestamps are embedded using the same positional encoding scheme and combined with a task embedding indicating the prediction task type.

The transformer encoder processes the embedded history sequence using multi-head attention. The architecture employs two transformer blocks, each containing a multi-head attention layer (4 heads, key dimension equal to the embedding size of 72) followed by two feed-forward layers with GELU activation. The first feed-forward layer expands to $4 \times 4 \times 72 = 1{,}152$ dimensions, while the second projects back to 72 dimensions. The target embedding is appended to the historical sequence, and the transformer attends over all positions including the target.

The final prediction is extracted from the target position's representation after transformer processing. A single dense layer with ReLU activation produces the output value, which is then rescaled by multiplying with the max-scaling factor: output = dense(transformer_output) $\times$ scale . This ensures predictions remain on the original scale of the input series.

3.2.3 Implementation Issues and Corrections

During implementation, several critical problems with the original ForecastPFN approach were identified. While some issues were successfully addressed, others persist and fundamentally affect the interpretation of results, as discussed in Section 4, below.

Scaling Instability: The original ForecastPFN employed a custom "robust scaler" that caused gradient explosion during training. Therefore, we replaced it with the simpler max-scaling.

Training Configuration: Stable training was only achieved through experimentation. Ultimately, we find that using the Adam optimiser with learning rate 0.0001 and batch size 1,024 produces stable training without numerial instability or training divergence. The loss function was corrected to operate on scaled values ($\mathcal{L} = \text{MSE}(y/\text{scale}, \hat{y}/\text{scale})$) to prevent series with large magnitudes from dominating gradient updates.

Evaluation Methodology: We discovered two fundamental evaluation problems in the original evaluation methodology. First, evaluation was performed on normalised data rather than original-scale data, violating established forecasting standards [8]. Second, with no obvious justification or explanation, evaluation was conducted on only a single randomly selected series per dataset rather than all series, making reported results unrepresentative. Thus, all evaluation reported here uses original-scale data across all series in each dataset, following best practices.

Persistent Issues

Prediction Flatness: Despite correcting the above issues, the model's predictions exhibit unusual "flatness": forecasts show minimal variation across time and fail to capture the dynamic patterns present in the input series. Visual inspection of predictions across multiple datasets reveals that the model produces forecasts that are excessively smooth and unresponsive to the variability in historical data. While this is sometimes justified, and indeed similar behaviour is sometimes observed in the predictions made by other models such as AutoETS, in this case the behaviour persists regardless of dataset characteristics, suggesting a fundamental architectural limitation rather than a data-specific problem.

The cause of this problem remains unclear but likely stems from one or more of the following: (a) the transformer architecture with only two encoder blocks may lack sufficient capacity to capture complex temporal dynamics from LGT-simulated data, (b) architectural mismatch between the design and the characteristics of LGT simulations, or (c) the training objective (MSE on scaled

values) may inadvertently penalise dynamic predictions in favour of safe, flat forecasts. It is beyond the scope of this work to fully resolve these issues, and we leave this for future work.

The unresolved problem of "flat" predictions is the most significant confounding factor in assessing whether LGT works effectively as a prior for PFN training. Poor quantitative performance may reflect these architectural deficiencies rather than limitations of LGT-simulated training data. The persistence of flat predictions even after correcting training, scaling, and evaluation issues also raises serious questions about the validity of results reported in the original ForecastPFN paper, as these architectural limitations would affect any training data source. However, by identifying and correcting these issues, we have laid the foundation for future research concerning the application of PFN architectures to forecasting within a rigorous and scientifically valid framework.

3.2.4 Input and Output Format

The model accepts fixed-length time series history along with corresponding timestamps. Each training example consists of: (1) a historical sequence of values, (2) timestamps for the historical period encoded as year, month, day, and day of week, (3) a target timestamp indicating when to forecast, and (4) a task indicator. The model outputs a single point forecast for the specified target timestamp.

During training, the model learns to predict random points in the future up to a maximum forecast horizon given historical data. This random target selection during training, as well as combining with the diverse synthetic series generated by LGT, enables the model to learn flexible forecasting strategies rather than overfitting to specific forecast horizons or series characteristics.

3.3 Training Details

3.3.1 Training Data and Preparation

We train the model on the 142,800 synthetic time series generated from LGT-fitted M3 monthly models, as described in the previous section. Each synthetic series has length 200, providing substantial temporal context for learning forecasting patterns. The training process uses TensorFlow's data pipeline with shuffling to ensure the model encounters diverse series patterns throughout training, preventing it from learning spurious correlations based on the order of presentation.

The synthetic data is processed into training examples through a framing

procedure. For each series, overlapping windows of fixed length are extracted, with each window serving as a potential training example. Within each window, the model learns to predict a randomly selected target point given the preceding history. This random target selection during training, which varies the forecast horizon across examples, encourages the model to learn flexible forecasting strategies rather than specialising for a particular lead time.

3.3.2 Loss Function and Optimisation

We use the Mean Squared Error (MSE) as the primary loss function in training. The model is optimised using the Adam optimiser [36] with a learning rate of 0.0001. This fixed learning rate is maintained throughout training rather than employing a schedule, as preliminary experiments showed stable convergence with this configuration.

An important detail of the loss computation involves the scaling procedure. As described in the architecture section, input series are scaled by their maximum values during embedding. To ensure the loss function operates on comparable scales across series with different magnitudes, both predictions and targets were divided by the scaling factor before computing MSE: $\mathcal{L} = \text{MSE}(y/\text{scale}, \hat{y}/\text{scale})$. This scaled loss prevents series with large absolute values from dominating the gradient updates.

3.3.3 Training Configuration

We train the model with 300 epochs with 1,000 steps per epoch, where each step processes a batch of 1,024 examples. The batch size of 1,024 was chosen to balance training stability with computational efficiency on available GPU hardware. Throughout training, multiple evaluation metrics are computed: Mean Absolute Percentage Error (MAPE), Mean Squared Error (MSE), and Symmetric Mean Absolute Percentage Error (sMAPE). These metrics are evaluated on both the training batches and a held-out validation set. The validation set consists of examples extracted from a separate slice of the synthetic data, cached for efficient repeated evaluation at the end of each epoch.

3.4 Inference and Prediction Process

Following the ForecastPFN approach, the model produces point forecasts at inference time. Critical corrections to the evaluation methodology are discussed in Section 3.2.3, including evaluation on original-scale rather than normalised data, and comprehensive evaluation across all series rather than random samples.

The inference process generates multiple overlapping windows from the test series history. Through empirical investigation, we discovered that using the window closest to the last observed point yields superior prediction performance compared to averaging across all windows, which is the default ForecastPFN approach. Therefore, this modified inference strategy is adopted: for each forecast horizon, the most recent window's prediction rather than averaging predictions from multiple windows is used. This modification ensures our predictions maximally leverage the most recent available information while avoiding effects from averaging over windows that include older, less relevant data.

4 Results and Discussion

4.1 Setup

4.1.1 Datasets

We evaluate our model on five benchmark datasets to assess both in-domain performance and generalisation to new domains. The primary evaluation uses the M3 monthly dataset (1,428 series, 18-month horizon) [15], which shares the same source as the LGT training data but relies on held-out test portions unseen during model development. To examine generalisation, four additional datasets are included: M3 yearly (645 series, 6-year horizon) to test performance at a different temporal frequency, and three datasets from the Monash Time Series Forecasting Repository [37], which are CIF 2016, FRED-MD, and Hospital, each evaluated with a 12-month forecast horizon following standard protocols. This design assesses whether patterns learned from LGT-simulated M3 data transfer effectively to both the original M3 test sets and distinct forecasting domains.

4.1.2 Baseline Methods

The predictions of our model are compared against three standard automatic forecasting methods: seasonal naive, Auto-ETS, and Auto-ARIMA. These baselines represent commonly used approaches that adapt automatically to different time series characteristics without manual tuning, providing appropriate benchmarks for evaluating zero-shot forecasting performance.

4.1.3 Evaluation Metrics

As discussed in previous sections, we evaluate forecast accuracy using the two scale-independent metrics and compute all metrics on the original, unscaled data to ensure meaningful interpretation and comparability across series with different magnitudes. In addition to forecast accuracy, the average prediction time per series is reported to assess computational efficiency. This provides insight into the practical feasibility of each method for operational forecasting applications, where both accuracy and speed matter.

4.2 Main Results

We present the results by comparing ForecastPFN trained with LGT-simulated data against three baseline methods: AutoETS, AutoARIMA, and Seasonal Naive. We evaluate the performance by using MASE (Table 1) and sMAPE (Table 2) across five datasets: M3 Monthly, M3 Yearly, CIF 2016, FRED-MD, and Hospital. Table 3 reports average prediction time per series for computational efficiency comparison.

Table 1: Mean Absolute Scaled Error (MASE) by dataset and method. Lower values indicate better performance. Best results in bold.

Dataset (horizon)	Our Model	AutoETS	AutoARIMA	Seasonal Naive
M3 Monthly (18)	1.75	**1.06**	1.14	1.37
M3 Yearly (6)	2.60	**2.13**	2.39	2.46
CIF 2016 (12)	**2.11**	2.45	2.39	2.43
FRED-MD (12)	4.01	**2.17**	2.30	6.20
Hospital (12)	1.07	**0.89**	0.90	1.08

Table 2: Symmetric Mean Absolute Percentage Error (sMAPE) by dataset and method. Lower values indicate better performance. Best results in bold.

Dataset (horizon)	Our Model	AutoETS	AutoARIMA	Seasonal Naive
M3 Monthly (18)	15.06	**14.16**	15.20	17.24
M3 Yearly (6)	18.50	**16.19**	16.76	17.88
CIF 2016 (12)	18.46	**13.62**	17.03	18.87
FRED-MD (12)	11.28	10.44	**10.43**	14.59
Hospital (12)	20.47	**17.68**	17.81	21.03

Table 3: Average prediction time per series in seconds.

Dataset	Our Model	AutoETS	AutoARIMA	Seasonal Naive
M3 Monthly	0.52	0.09	0.99	0.003
M3 Yearly	0.22	0.005	0.12	0.002
CIF 2016	0.28	0.07	0.52	0.003
FRED-MD	2.11	0.58	5.50	0.003
Hospital	0.22	0.07	0.59	0.003

4.2.1 Overall Performance Patterns

Table 1 shows MASE results across all datasets and methods. Our model achieves the lowest MASE on only one dataset (CIF 2016, MASE=2.11), while AutoETS achieves the best performance on the remaining four datasets. The magnitude of performance differences varies considerably: on some datasets our model's MASE approaches the baseline methods (Hospital: 1.07 vs. AutoETS 0.89, a difference of 0.18), while on others the gap is substantial (M3 Monthly: 1.75 vs. AutoETS 1.06, a difference of 0.69; FRED-MD: 4.01 vs. AutoETS 2.17, a difference of 1.84). Notably, on M3 Monthly, the primary evaluation dataset and the source of data used to condition LGT simulation. Our model trails not only AutoETS and AutoARIMA but also the simple Seasonal Naive baseline (1.37).

Table 2 presents sMAPE results, which are broadly consistent with the MASE findings but reveal some nuances. AutoETS again dominates, achieving the lowest sMAPE on four of five datasets. However, the relative performance gaps differ somewhat from MASE: on CIF 2016, our model achieves competitive MASE (2.11, winning outright) but substantially worse sMAPE (18.46 vs. AutoETS 13.62, a difference of 4.84). Conversely, on FRED-MD, our model shows relatively stronger sMAPE performance (11.28 vs. AutoETS 10.44, a difference of only 0.84) despite poor MASE performance (4.01 vs. 2.17), across most datasets, our model outperforms Seasonal Naive on sMAPE, suggesting it captures some useful patterns beyond naive seasonality.

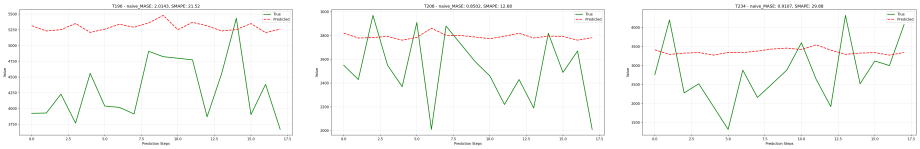
Table 3 shows prediction times. Our model requires 0.22-0.52 seconds per series on most datasets, faster than AutoARIMA (0.12-0.99 seconds) but slower than AutoETS (0.005-0.09 seconds) and much slower than Seasonal Naive (0.002-0.003 seconds). The exception is FRED-MD, where our model takes 2.11 seconds per series, which is substantially slower than on other datasets and comparable to AutoARIMA's 5.50 seconds. This computational anomaly coincides with FRED-MD being our worst-performing dataset by MASE, suggesting the model struggles with these macroeconomic series' characteristics.

It is worth noting that while inference times are slower than AutoETS and Seasonal Naive, the ForecastPFN architecture contains significantly more parameters and has been exposed to a substantially larger volume of training data

compared to the lightweight statistical baselines. If future refinements achieve competitive accuracy, these longer inference times may be justified by improved performance and the practical advantage of zero-shot deployment without requiring per-series model fitting or selection.

4.2.2 M3 Monthly and Yearly Performance

On M3 Monthly, the dataset used to condition LGT simulation for generating training data, the model achieves MASE of 1.75 compared to AutoETS (1.06), AutoARIMA (1.14), and Seasonal Naive (1.37). This 65% higher error than AutoETS is particularly concerning given that synthetic training data was generated specifically from LGT models fitted to M3 monthly training series. The model should theoretically have an advantage on this dataset, yet it performs worse than even the simple Seasonal Naive baseline. The sMAPE results (15.06) show modest improvement over Seasonal Naive (17.24) but remain worse than AutoETS (14.16) and comparable to AutoARIMA (15.20). Visual inspection of predictions on M3 Monthly series below (Figure 3, also see Appendix A.2) reveals that while the model captures some dynamic patterns, the predictions still exhibit less variability than the actual series, though not as severely as on other datasets.

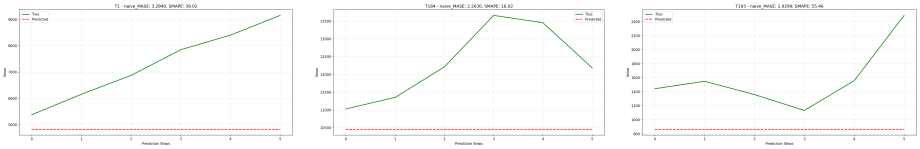


(a) T196 (b) T208 (c) T234

Figure 3: Sample forecasts on M3 Monthly series showing moderate pattern capture and limited variability.

On M3 Yearly, the model achieves MASE of 2.60, again trailing all baselines including Seasonal Naive (2.46). Since the model was trained exclusively on monthly data, some performance degradation on yearly data might be expected due to frequency mismatch. However, the consistent underperformance across both frequencies suggests deeper issues beyond frequency mismatch. Critically, M3 Yearly exhibits the severe prediction "flatness" problem: the model produces forecasts that are excessively smooth and fail to capture the volatility and turning points present in the actual yearly series. Sample predictions below in Figure 4 and in Appendix A.3 demonstrate this flatness clearly, with forecasts appearing as nearly horizontal lines while actual series exhibit substantial variation. This architectural limitation confounds assessment of whether the poor performance reflects inadequate learning from LGT-simulated data, failure to generalise to yearly frequency, or simply the model's inability to produce

dynamic predictions. The flatness is more pronounced on yearly data than monthly data, possibly because yearly series in M3 exhibit higher volatility relative to their length, exposing the model's tendency toward overly conservative predictions. If the model had learned robust forecasting principles from LGT simulations, better generalisation to different frequencies would be expected even with this architectural constraint.



(a) T1 (b) T184 (c) T193

Figure 4: Sample forecasts on M3 Yearly series showing severe flatness and under-variability compared to actual data.

4.2.3 Monash Repository Datasets Performance

Performance on the three Monash datasets reveals considerable heterogeneity, suggesting our model's effectiveness depends heavily on specific data characteristics.

CIF 2016 represents our model's only clear success: MASE of 2.11 outperforms AutoETS (2.45), AutoARIMA (2.39), and Seasonal Naive (2.43). This demonstrates the model can generalise beyond M3 and achieve competitive accuracy when data characteristics align favourably. However, the sMAPE results (18.46) tell a different story, with our model substantially trailing AutoETS (13.62) and AutoARIMA (17.03). This metric discrepancy,winning on MASE but losing substantially on sMAPE, suggests the model may handle certain error patterns better than others, or that the two metrics capture different aspects of forecast quality on this dataset.

FRED-MD presents a major failure case. Our model's MASE of 4.01 substantially exceeds AutoETS (2.17), AutoARIMA (2.30), and even Seasonal Naive (6.20). The prediction time of 2.11 seconds per series, which is ten times slower than on other monthly datasets. This indicates the model struggles computationally with these macroeconomic series. FRED-MD contains complex economic indicators with potentially different dynamics than the business/demographic series dominating M3. The poor performance suggests LGT-simulated training data, conditioned on M3 patterns, fails to capture the characteristics of macroeconomic forecasting problems.

Hospital data presents a contrasting picture: our model achieves MASE of 1.07, approaching AutoETS (0.89) and AutoARIMA (0.90) within 0.18 and

0.17 respectively. This near-competitive performance, combined with fast prediction times (0.22 seconds), suggests LGT-generated training data successfully captures patterns relevant to hospital demand forecasting. This is notable because hospital demand differs substantially from M3's business and economic series, yet the model generalises reasonably well. The sMAPE results (20.47 vs. AutoETS 17.68) show a larger gap, but overall Hospital represents a partial success case where the PFN approach nearly matches traditional methods.

4.3 Interpretation, Limitations, and Implications

The results demonstrate that training ForecastPFN on LGT-simulated data produces a model that generally underperforms traditional automatic forecasting methods, though with notable exceptions. However, multiple confounding factors make it difficult to isolate whether this reflects limitations of LGT as a prior or fundamental issues with the ForecastPFN architecture itself. The key limitations are: (a) fundamental issues with the ForecastPFN architecture and implementation, (b) limited diversity in training data from a single source dataset, (c) potential architectural mismatch with LGT-simulated data characteristics, and (d) challenges in transforming LGT from forecaster to data generator.

4.3.1 ForecastPFN Implementation Issues: The Primary Confounding Factor

The most significant issue is that multiple critical problems with the original ForecastPFN implementation were discovered during this work. These include training instabilities, improper scaling causing gradient explosion, evaluation on normalised data violating forecasting standards, evaluation on single random series rather than complete datasets, and persistent prediction flatness. While several issues were corrected, including implementing proper scaling, evaluating on original-scale data across all series, and stabilising training, the predictions still exhibit unusual characteristics.

This is the biggest factor clouding results: the persistent flatness and poor performance even after fixing known problems strongly suggest (a) the ForecastPFN architecture's limitations confound the ability to assess LGT as a prior, and (b) the results in the original ForecastPFN paper are highly suspect. If the base architecture cannot produce reasonable forecasts even with corrected procedures, then poor performance may reflect architectural deficiencies rather than limitations of LGT-simulated training data. These findings call into question both prior PFN forecasting claims and the suitability of this architecture for learning from any prior distribution.

4.3.2 Additional Limiting Factors

Limited Training Data Diversity: Training exclusively on simulated data from M3 monthly series yields 142,800 series, but all derive from parameter distributions learned from the same 1,428 M3 monthly series. The M3 dataset, though diverse, represents only a subset of possible time series behaviours. More diverse training incorporating multiple source datasets across domains and frequencies might improve generalisation, though this cannot be assessed until architectural issues are resolved.

Architectural Mismatch with LGT Characteristics: The ForecastPFN transformer (two encoder layers, multi-head attention) was developed for Gaussian Process synthetic data with multiplicative components. Whether this architecture suits LGT simulations encoding power-law trends, Student-t errors, and heteroscedastic variance remains unclear. The underwhelming results may reflect architectural mismatch rather than LGT limitations. Alternative architectures designed for LGT's structural assumptions might perform better, but separating architectural from prior-related issues proves difficult given the implementation problems encountered.

LGT as Forecaster vs. Data Generator: Using LGT to generate neural network training data introduces challenges. When forecasting directly, LGT leverages fitted parameters tailored to each specific series. When generating training data, parameters are sampled from pooled distributions and series are simulated from randomised initial conditions. This may not preserve subtle patterns making LGT effective as a direct forecaster. Simulated data may capture general structural assumptions but not detailed patterns enabling accurate real-series forecasts.

4.3.3 Implications and Partial Successes

Despite these limitations, results are not uniformly negative. Success on CIF 2016 and near-competitive performance on Hospital demonstrate LGT-simulated data can capture relevant patterns for certain domains. Consistent sMAPE improvement over Seasonal Naive suggests the model learns beyond naive seasonality. These partial successes indicate the core approach has merit, even if current implementation falls short on most benchmarks.

However, given the multiple confounding factors, particularly the fundamental architectural issues, current results do not definitively assess whether LGT works as a prior for PFN training. The persistent prediction anomalies after corrections suggest the ForecastPFN architecture itself may be fundamentally limited, making it impossible to separate architectural from prior-related issues. Future work requires either substantial architectural modifications or entirely

different neural network designs before LGT's effectiveness as a prior can be properly evaluated.

4.4 Future Directions

Several clear paths forward emerge from this work. Most critically, the architectural and implementation issues we identified must be resolved before LGT's effectiveness as a prior can be properly assessed. This requires either substantial modifications to ForecastPFN or exploration of entirely different neural architectures designed specifically for LGT-simulated data characteristics.

Beyond architectural improvements, incorporating diverse source datasets across multiple domains and frequencies when learning LGT parameter distributions could enrich training data diversity. The current restriction to M3 monthly data limits the range of patterns in simulated training series.

Finally, extending the approach to generate probabilistic forecasts rather than point predictions would better leverage LGT's natural ability to produce full predictive distributions through its Bayesian formulation. Ensemble approaches combining PFN predictions with traditional methods may also leverage complementary strengths.

5 Conclusion

This research investigated whether the Robust Local and Global Trend (LGT) model could serve as an effective prior for training Prior-Fitted Networks, enabling zero-shot time series forecasting. By fitting LGT to M3 monthly series, generating 142,800 synthetic time series from the learned parameter distributions, and training a ForecastPFN architecture on this synthetic data, the study explored a novel approach to creating generalisable forecasting models without requiring training on real data from target domains.

The empirical results demonstrate that whilst the current implementation underperforms traditional automatic methods such as AutoETS and AutoARIMA on most datasets, the approach shows promising signals. The model achieves competitive performance on specific datasets, notably CIF 2016 and Hospital, and consistently outperforms seasonal naive on sMAPE metrics. These successes demonstrate that LGT-generated synthetic data can capture meaningful forecasting patterns, validating the core premise that Bayesian exponential smoothing models can serve as priors for neural network training.

The challenges encountered illuminate important research directions rather

than fundamental limitations. The restriction to M3 monthly data for learning LGT parameters represents an artificial constraint; incorporating diverse datasets across multiple domains and frequencies when generating synthetic training data offers clear potential for improvement. Similarly, the current study examined only one architecture and one prior model, leaving vast design space unexplored. Alternative architectures specifically designed for LGT-generated data characteristics, or ensemble approaches combining multiple prior models, may unlock substantially better performance. By systematically documenting the specific shortcomings observed in this implementation, from insufficient synthetic data diversity to potential architecture-prior misalignment, this work provides a clear methodological roadmap for advancing PFN-based forecasting research with improved rigour and scientific validity.

The zero-shot forecasting approach embodied by PFNs addresses a fundamental challenge in time series analysis: the need for models that generalise across domains without requiring extensive training data from each new application. While traditional methods like AutoETS remain effective, they rely on decades of statistical theory development and careful engineering. The PFN approach offers a complementary path: encoding statistical knowledge through synthetic data generation rather than explicit algorithmic design. The current results establish a foundation and identify clear paths forward: more diverse synthetic data, alternative prior models, architectural innovations, and probabilistic forecasting extensions.

As forecasting applications expand into new domains with limited historical data, zero-shot approaches become increasingly valuable. This research demonstrates both the potential and the challenges of using Bayesian statistical models as priors for neural forecasting. The path forward requires not abandoning the approach, but refining it through more diverse training data, better alignment between prior models and architectures, and continued exploration of the rich design space that the PFN paradigm offers.

6 Ethics Statement

This research uses publicly available time series datasets from the M3 forecasting competition and the Monash Time Series Forecasting Repository. No human subjects were involved, and no ethical concerns have been identified.

7 References

[1] G. Mahalakshmi, S. Sridevi, and S. Rajaram, "A survey on forecasting of time series data," in *2016 International Conference on Computing Technologies and Intelligent Data Engineering (ICCTIDE'16)*, Kovilpatti, India: IEEE, Jan. 2016, pp. 1–8, ISBN: 978-1-4673-8437-7. DOI: 10.1109/ICCTIDE.2016.7725358. (visited on 03/23/2025).

[2] G. E. P. Box and G. M. Jenkins, *Time Series Analysis; Forecasting and Control* (Holden-Day Series in Time Series Analysis). San Francisco: Holden-Day, 1970.

[3] C. C. Holt, "Forecasting seasonals and trends by exponentially weighted moving averages," *International Journal of Forecasting*, vol. 20, no. 1, pp. 5–10, 2004, ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2003.09.015.

[4] R. J. Hyndman and Y. Khandakar, "Automatic Time Series Forecasting: The forecast Package for R," *Journal of Statistical Software*, vol. 27, no. 3, pp. 1–22, 2008. DOI: 10.18637/jss.v027.i03.

[5] R. J. Hyndman, A. B. Koehler, R. D. Snyder, and S. Grose, "A state space framework for automatic forecasting using exponential smoothing methods," *International Journal of Forecasting*, vol. 18, no. 3, pp. 439–454, 2002, ISSN: 0169-2070. DOI: 10.1016/S0169-2070(01)00110-8.

[6] K. Benidis *et al.*, "Deep learning for time series forecasting: Tutorial and literature survey," *ACM Computing Surveys*, vol. 55, no. 6, pp. 1–36, Dec. 2022, ISSN: 1557-7341. DOI: 10.1145/3533382.

[7] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "Statistical and Machine Learning forecasting methods: Concerns and ways forward," *PLoS ONE*, vol. 13, Mar. 2018. DOI: 10.1371/journal.pone.0194889.

[8] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *International Journal of Forecasting*, vol. 22, no. 4, pp. 679–688, 2006, ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2006.03.001.

[9] F. Petropoulos *et al.*, "Forecasting: Theory and practice," *International Journal of Forecasting*, vol. 38, no. 3, pp. 705–871, 2022, ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2021.11.001.

[10] S. Müller, N. Hollmann, S. P. Arango, J. Grabocka, and F. Hutter, "Transformers can do bayesian inference," *arXiv preprint arXiv:2112.10510*, 2021. arXiv: 2112.10510.

[11] N. Hollmann, S. Müller, K. Eggensperger, and F. Hutter, "Tabpfn: A transformer that solves small tabular classification problems in a second," *arXiv preprint arXiv:2207.01848*, 2022. arXiv: 2207.01848.

[12] S. Dooley, G. S. Khurana, C. Mohapatra, S. V. Naidu, and C. White, "ForecastPFN: Synthetically-Trained Zero-Shot Forecasting," in *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., vol. 36, Curran Associates, Inc., 2023, pp. 2403–2426.

[13] S. Smyl, C. Bergmeir, A. Dokumentov, X. Long, E. Wibowo, and D. Schmidt, "Local and global trend Bayesian exponential smoothing models," *International Journal of Forecasting*, vol. 41, no. 1, pp. 111–127, 2025, ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2024.03.006.

[14] X. Long, D. F. Schmidt, C. Bergmeir, and S. Smyl, "Fast Gibbs sampling for the local-seasonal-global trend Bayesian exponential smoothing model," *Statistics and Computing*, vol. 35, no. 3, p. 77, Apr. 2025, ISSN: 1573-1375. DOI: 10.1007/s11222-025-10603-z.

[15] S. Makridakis and M. Hibon, "The M3-Competition: Results, conclusions and implications," *International Journal of Forecasting*, vol. 16, no. 4, pp. 451–476, 2000, ISSN: 0169-2070. DOI: 10.1016/S0169-2070(00)00057-1.

[16] R. Hyndman, "3.8 another look at forecast-accuracy metrics for intermittent demand," *Business Forecasting: Practical Problems and Solutions*, p. 204, 2015.

[17] L.-R. Forecasting, "From crystal ball to computer," *Scott Armstrong Robert J. Genetski*, 1978.

[18] R. J. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*. OTexts, 2018.

[19] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*. John Wiley & Sons, 2015.

[20] R. Hyndman, A. Koehler, K. Ord, and R. Snyder, *Forecasting with Exponential Smoothing: The State Space Approach*. Springer, 2008.

[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735.

[22] A. Vaswani *et al.*, "Attention is all you need," 2017.

[23] B. Lim and S. Zohren, "Time-series forecasting with deep learning: A survey," *Philosophical Transactions of the Royal Society A*, vol. 379, no. 2194, p. 20 200 209, 2021.

[24] L. J. Tashman, "Out-of-sample tests of forecasting accuracy: An analysis and review," *International Journal of Forecasting*, vol. 16, no. 4, pp. 437–450, 2000, ISSN: 0169-2070. DOI: 10.1016/S0169-2070(00)00065-0.

[25] S. Brooks, "Markov chain Monte Carlo method and its application," *Journal of the royal statistical society: series D (the Statistician)*, vol. 47, no. 1, pp. 69–100, 1998.

[26] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, pp. 859–877, 2017. DOI: 10.1080/01621459.2017.1285773. eprint: https://doi.org/10.1080/01621459.2017.1285773.

[27] T. Nagler, "Statistical foundations of prior-data fitted networks," in *International Conference on Machine Learning*, PMLR, 2023, pp. 25 660–25 676.

[28] S. B. Hoo, S. Müller, D. Salinas, and F. Hutter, "The tabular foundation model TabPFN outperforms specialized time series forecasting models based on simple features," in *NeurIPS Workshop on Time Series in the Age of Large Models*, 2024.

[29] P. J. Harrison and C. F. Stevens, "Bayesian forecasting," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 38, no. 3, pp. 205–228, Dec. 2018, ISSN: 0035-9246. DOI: 10.1111/j.2517-6161.1976.tb01586.x. eprint: https://academic.oup.com/jrsssb/article-pdf/38/3/205/49116951/jrsssb_38_3_205.pdf.

[30] G. M. Martin *et al.*, "Bayesian forecasting in economics and finance: A modern review," *International Journal of Forecasting*, vol. 40, no. 2, pp. 811–839, 2024, ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2023.05.002.

[31] K. L. Lange, R. J. A. Little, and J. M. G. Taylor, "Robust statistical modeling using the t distribution," *Journal of the American Statistical Association*, vol. 84, no. 408, pp. 881–896, 1989. DOI: 10.1080/01621459.1989.10478852. eprint: https://doi.org/10.1080/01621459.1989.10478852.

[32] E. O. Taga, M. E. Ildiz, and S. Oymak, "TimePFN: Effective Multivariate Time Series Forecasting with Synthetic Data," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 19, pp. 20 761–20 769, Apr. 2025. DOI: 10.1609/aaai.v39i19.34288. (visited on 10/06/2025).

[33] H. Zhou *et al.*, *Informer: Beyond efficient transformer for long sequence time-series forecasting*, 2021. arXiv: 2012.07436 [cs.LG].

[34] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting," in *Proceedings of the 39th International Conference on Machine Learning*, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., ser. Proceedings of Machine Learning Research, vol. 162, PMLR, Jul. 2022, pp. 27 268–27 286.

[35] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The M4 Competition: Results, findings, conclusion and way forward," *International Journal of Forecasting*, vol. 34, no. 4, pp. 802–808, 2018, ISSN: 0169-2070. DOI: 10.1016/j.ijforecast.2018.06.001.

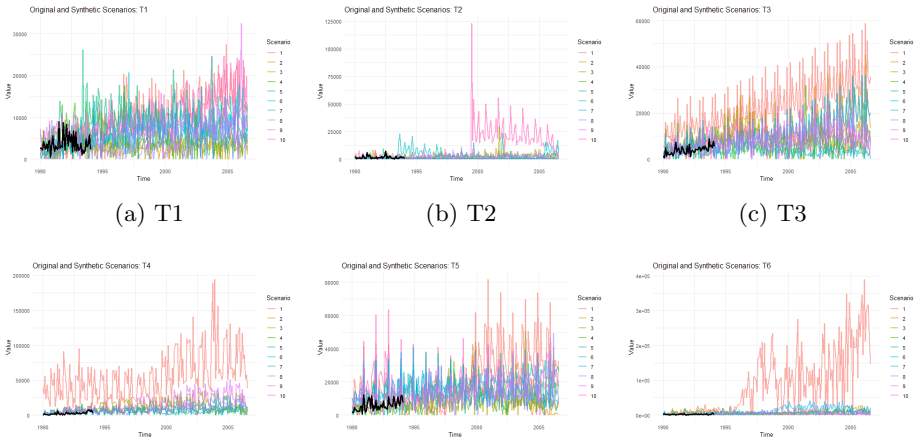
[36] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2017. arXiv: 1412.6980 [cs.LG].

[37] R. Godahewa, C. Bergmeir, G. I. Webb, R. J. Hyndman, and P. Montero-Manso, "Monash time series forecasting archive," in *Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021.

Appendix

A.1 Sample Series

Here are the generated sample series; only 10 are displayed for each plot for easier visualisation. The black line is the original series.



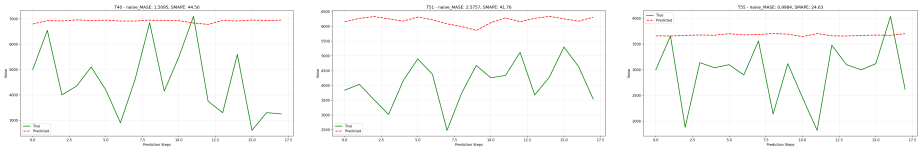
(a) T1 (b) T2 (c) T3

(d) T4 (e) T5 (f) T6

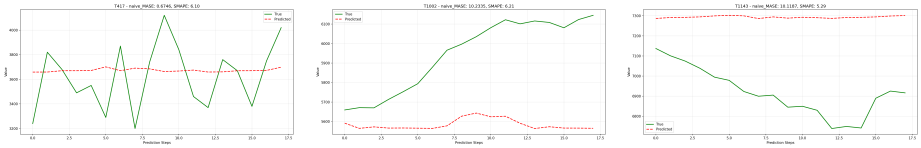
Figure 5: Examples of generated sample series. Only 10 samples per plot are displayed for clearer visualisation.

A.2 M3 Monthly Prediction Examples

Figures 6 show sample predictions on M3 Monthly series. While the model exhibits some ability to track trends and seasonal patterns, predictions display less variability than the actual series. The flatness issue is present but moderate compared to yearly data.



(a) T40 (b) T51 (c) T55

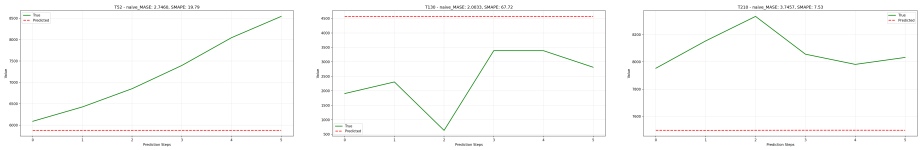


(d) T417 (e) T1002 (f) T1143

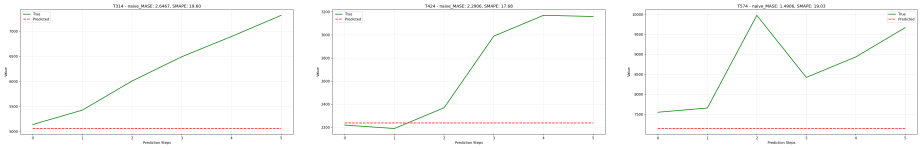
Figure 6: Example M3 Monthly actual vs. predicted series. Each subplot corresponds to one sample series.

A.3 M3 Yearly Prediction Examples

Figures 7 demonstrate the severe prediction flatness observed on M3 Yearly series. Forecasts appear as nearly horizontal or minimally varying lines while actual series exhibit substantial volatility and turning points. This pronounced flatness significantly impacts forecast accuracy and demonstrates the architectural limitations discussed.



(a) T52 (b) T130 (c) T210



(d) T314 (e) T424 (f) T574

Figure 7: Example M3 Yearly actual vs. predicted series. The predictions show strong flatness relative to actual series.