University of Geneva

Master of Science in Statistics

Time Series Engression

Modeling Conditional Distributions with Input-Dependent Noise and Temporal Structure

---

Master's Thesis

Author

Yuri Croci

yuri.croci@etu.unige.ch

Supervisor

Prof. Dr. Sebastian Engelke

sebastian.engelke@unige.ch

Geneva, 14 November 2025

Abstract

Temporal forecasting of rare and extreme events requires reliable uncertainty quantification through modeling full conditional distributions, yet most approaches rely on restrictive parametric assumptions that limit their flexibility in capturing complex real-world dynamics. This thesis extends Engression, a pre-additive noise framework originally developed for static settings, to time series applications through the integration of temporal encoders and heteroskedastic noise modeling. This approach enables learning of conditional distributions that capture both temporal dependencies and input-dependent variability while preserving the extrapolation stability offered by the pre-additive design. The proposed framework was evaluated through a controlled simulation study and a real-world river discharge forecasting application, with systematic assessment of performance stability across multiple random initializations. Results demonstrate that Engression-based models achieve competitive mean prediction performance alongside superior extrapolation capabilities compared to their deterministic counterparts. While the proposed individual extensions provided context-dependent benefits, their combination proved effective in capturing complex volatility patterns, particularly improving tail quantile estimation in asymmetric distributions. The findings establish this extended framework as a viable approach for distributional temporal forecasting, enabling reliable uncertainty quantification and enhanced extreme event prediction without parametric constraints.

Contents

1 Introduction

1.1 Challenges in Forecasting Rare and Extreme Events

In many practical domains, such as finance and environmental risk management, critical decisions often depend on the ability to forecast rare but high-impact events. Catastrophes like stock market crashes or severe floods, although infrequent, can cause substantial damage and often deviate from historical patterns, making them especially difficult to anticipate. Consequently, an increased demand for forecasting models that provide comprehensive uncertainty quantification has emerged. However, in practice, this need remains largely unmet, as most traditional forecasting models continue to focus on minimizing point prediction error, typically by estimating the conditional mean. While efficient at summarizing central tendencies, such models offer little insight into the likelihood of rare outcomes. To address this shortcoming, recent studies have shifted focus toward learning the full conditional distribution, which provides a more expressive characterization of the predictive uncertainty. Although this alternative offers a richer representation of uncertainty than point forecasts, it also introduces considerable complexity, both conceptually and computationally. As a result, most models still rely on simplifying assumptions such as fixed parametric forms or homoscedastic noise structure, limiting their adaptability and effectiveness in practice.

A second major limitation concerns the difficulty of generalizing to previously unseen or evolving patterns, particularly those associated with extreme events. As models become more flexible to better capture data variability, they often face a trade-off between expressiveness and robustness. When leaning too far toward the former, models sharply increase their overfitting to the observed training data, degrading their performance in out-of-distribution scenarios. This issue becomes critical when the phenomena of interest lie in the distribution tails, making them inherently rare in the observed data. Such a trade-off is particularly evident in natural processes, where shifts in patterns due to climate change have increased the frequency and intensity of extreme events, generating patterns that deviate from past observations. Models that cannot adapt to such changes may yield unreliable predictions precisely when accurate forecasting is most essential for long-term planning and early intervention.

In light of these limitations, researchers have increasingly explored modeling strategies that aim to quantify uncertainty more reliably. This work advances these efforts by proposing a distributional approach for temporal forecasting, offering improved robustness in uncertainty quantification without relying on restrictive parametric assumptions.

1.2 Pre-Additive Noise and the Promise of Engression

A classical paradigm in statistical modeling concerns how randomness is introduced into a model. Among these, one of the most common approaches is to treat noise as an additive perturbation to a deterministic signal, leading to the post-additive formulation

$$Y = f(X) + \varepsilon, \quad \varepsilon \sim \mathcal{D}, \quad \varepsilon \perp\!\!\!\perp X, \tag{1}$$

where Y denotes the target variable to be predicted, X the associated predictors, f a deterministic function, and ε a noise term drawn from a fixed Gaussian distribution $\mathcal{D} = \mathcal{N}(0, \sigma^2)$, assumed to be independent of X. This formulation is popular in practice due to its simplicity and compatibility with standard estimation methods such as least squares, offering a solid foundation for predictive modeling.

Although widely used, this approach relies on two strong assumptions: that the covariates are assumed to be measured without error and that randomness affects only the response, not the inputs themselves. In many real-world scenarios, however, covariates may be noisy, shaped by unobserved processes, or represent latent stochastic systems. As a result, observed covariates often serve only as a noisy proxy for the true underlying factors, making the assumption of independence between X and ε statistically unlikely. When this independence is violated, the model tends to suffer from statistical misspecification, leading to biased estimates and unreliable inference.

Motivated by the shortcomings of the post-additive framework, the pre-additive formulation offers an alternative perspective on uncertainty. Instead of injecting noise after a deterministic transformation, it models randomness through input-level interactions

$$Y = f(X, \eta), \quad \eta \sim \mathcal{D}, \quad \eta \perp\!\!\!\perp X, \tag{2}$$

where η is sampled independently of X from a distribution $\mathcal{D}$. In doing so, the model allows uncertainty to propagate directly through the nonlinear transformation f. As a result, the output distribution can reflect the full effect of variability in the inputs, rather than modeling symmetric residual noise around a fixed signal as in the post-additive case.

Due to this structure, the pre-additive approach serves as a useful baseline for generative modeling in complex data, as it aligns with many real-world systems where covariates are subject to uncertainty. Furthermore, by allowing the noise to propagate through the function f, the models using this formulation are able to capture input-driven variability without relying on fixed likelihoods or parametric assumptions. Instead, they learn the

conditional distribution $P(Y \mid X)$ implicitly through direct sampling of η, making them well-suited for flexible and data-driven predictive tasks in modern machine learning.

Among the models that adopt this philosophy, Engression (Shen and Meinshausen, 2024) provides a promising framework and serves as the foundation for this study. Specifically, Engression estimates the full conditional distribution $P(Y \mid X)$ by concatenating a noise vector $\eta \sim \mathcal{N}(0, \sigma^2 I)$ at the input level and passing the combined vector through a deterministic function $f(X, \eta)$ parameterized by a deep neural network. In doing so, the model approximates the conditional distribution by learning how input perturbations translate into output variability.

A key advantage of this design lies in its ability to extrapolate near the boundary of the training domain. As shown by Shen and Meinshausen (2024), this property holds under mild conditions on f, such as local monotonicity in x and bounded noise support, allowing for meaningful predictions even in extreme regions. Moreover, when these conditions are only partially met, the model still preserves stable extrapolation by reverting to linear behavior, offering robustness in more complex cases. While Engression has demonstrated strong performance in static settings, both in simulations and empirical applications, its extension to sequential or time-dependent contexts remains largely unexplored. Addressing this key limitation in the context of time-dependent modeling will be the central objective of the present thesis.

1.3 Extending Engression to Time Series Forecasting

Building on the formulations discussed above, we aim to extend the Engression framework to the more complex setting of time series forecasting. To achieve this goal, we introduce two key modifications that are specifically designed to account for the sequential nature of temporal data, and their potential heteroskedasticity.

The first modification introduces a temporal encoder that summarizes past observations over a fixed-length lag window. To construct this component, we rely on standard sequence modeling architectures, which transform the recent input history into a latent representation $\tilde{h}_t$. The resulting vector is then passed to the Engression decoder, which follows the same pre-additive sampling principles to generate the conditional distribution of the next-step forecast.

The second modification addresses the limitation of fixed, input-independent noise in the original Engression setup. To overcome this limitation, we introduce a dynamic noise injection mechanism, where noise is sampled from a zero-mean Gaussian distribution

with variance $\tilde{\sigma}^2(\tilde{h}_t)$, learned as a function of the temporal representation $\tilde{h}_t$. Through this adjustment, the model is designed to capture both time-varying behavior and input-dependent uncertainty, thereby improving its ability to produce accurate distributional forecasts in heteroskedastic environments.

Combined, these two extensions form a generative model capable of learning the full conditional distributions in dynamic systems by integrating temporal structure and input-driven variability. To evaluate its effectiveness, we implement a rigorous and reproducible benchmarking framework using both synthetic and real-world datasets. The evaluation is based on the model's ability to extrapolate beyond the training support and produce accurate distributional forecasts. Additionally, to isolate the contribution of each component, we test different combinations of the proposed modifications, including deterministic variants where pre-additive noise injection is removed.

1.4 Thesis Structure

The thesis is organized as follows.

Chapter 2 presents the theoretical foundations of the work, covering the history and technical aspects of distributional regression, the Engression framework, and recurrent neural network architectures for sequential data modeling.

Chapter 3 details the proposed extensions of Engression to the time series setting, including the integration of temporal encoders and heteroskedastic noise injection, along with the experimental design adopted for their evaluation.

Chapter 4 describes the design of the simulation study and applies the proposed frameworks to a synthetic time series with known ground truth dynamics, allowing for performance comparison in a controlled environment.

Chapter 5 presents the Swiss river discharge dataset and evaluates the proposed architectures in their ability to capture uncertainty and extrapolate beyond the training support.

Chapter 6 summarizes the main findings across both experimental settings, discusses the limitations of the proposed approaches and outlines potential directions for future research in distributional approaches for temporal forecasting.

Finally, **Chapter 7** synthesizes the key contributions and broader implications of this work.

2 Background

2.1 Distributional Regression and Proper Scoring Rules

Throughout the evolution of statistical modeling, different estimation tasks have gained attention, reflecting evolving scientific goals and increasing availability of technological resources. The foundations of predictive modeling were established by Galton (1886), with further formalization by Pearson and Henrici (1896), focusing on estimating the conditional mean $\mathbb{E}[Y \mid X]$ through linear relationships. Accompanying this, the mean squared error (MSE) emerged as the standard loss function, favored for its analytical practicality and its equivalence to maximum likelihood estimation under Gaussian assumptions.

As applications expanded to domains where tail behavior and asymmetric risk were critical, the limitations of mean-based regression became increasingly evident. To address this, Koenker and Bassett (1978) introduced quantile regression, which uses the pinball loss to estimate conditional quantiles $Q_\tau(Y \mid X = x)$ for any level $\tau \in (0, 1)$. Despite the absence of closed-form solutions for the pinball loss function, modern optimization methods have enabled the practical use of quantile regression, which continues to play an important role in applied modeling. However, while practically viable, a key limitation of this framework lies in the independent estimation of multiple quantiles. More specifically, it increases computational burden and introduces the risk of quantile crossing, complicating inference when full distributional understanding is needed.

More recently, supported by data availability and increased computational power, research efforts have shifted toward estimating the full conditional distribution $P(Y \mid X)$ rather than individual summaries (Gneiting and Katzfuss, 2014). This distributional perspective enables richer uncertainty quantification, which is especially valuable in fields where rare but impactful events matter as much as central tendencies. The general approach is typically parametric, where a likelihood family is defined over the observed data and its parameters are estimated via maximum likelihood, for example by using conditional variational autoencoders (VAEs, Sohn et al., 2015). The respective negative log-likelihood then serves as a scoring rule, a loss function that evaluates the quality of probabilistic forecasts by comparing the predicted distribution against observed outcomes. A scoring rule is further defined as proper if the expected score under the true distribution is minimized by issuing forecasts from that same distribution, thereby incentivizing realistic and calibrated uncertainty estimates.

In many modern problems, however, the conditional distribution is too complex or irregular to be well approximated by any fixed parametric family. This limitation has motivated the

development of flexible, simulation-based generative models that produce samples from $P(Y \mid X)$ directly, without relying on explicit likelihood functions, such as conditional generative adversarial networks (GANs, Mirza and Osindero, 2014). By avoiding such parametric restrictions, these models can represent more complex patterns and enable empirical estimation of quantities such as means, quantiles, or tail risks through direct sampling. To evaluate the quality of these predictive samples, proper scoring rules have been developed that compare them against observed outcomes. Among these, the Continuous Ranked Probability Score (CRPS) and the Energy Score have emerged as central tools (Gneiting and Raftery, 2007). While mathematically equivalent in the univariate case, the Energy Score extends naturally to multivariate targets and is strictly proper, making it suitable for a broader class of forecasting tasks, especially in multivariate and nonparametric settings.

Given the observed univariate target y_i for observation i, and a set of m predictive samples $\{\hat{y}_i^{(j)}\}_{j=1}^m$ indexed by j, the Energy Score is defined as

$$\mathrm{ES}\Big(y_i, \{\hat{y}_i^{(j)}\}_{j=1}^m\Big) = \frac{1}{m}\sum_{j=1}^{m}\left\|y_i - \hat{y}_i^{(j)}\right\| - \frac{1}{2m(m-1)}\sum_{j=1}^{m}\sum_{\substack{k=1\\k\neq j}}^{m}\left\|\hat{y}_i^{(j)} - \hat{y}_i^{(k)}\right\|, \tag{3}$$

where $\|\cdot\|$ denotes the Euclidean norm.

In this formulation, the first term encourages alignment with the observed value, causing the distribution to concentrate around the target response. The second term, by contrast, promotes dispersion by measuring the average pairwise distance between the predictive samples, preventing the model from collapsing to a point estimate. This balance between concentration and dispersion reflects the dual objective of distributional regression: producing sharp predictions while preserving well-calibrated uncertainty. Moreover, the Energy Score (and its negative counterpart, the energy loss) is differentiable and compatible with gradient-based optimization, making it a flexible and principled objective for training and evaluating generative models.

Given its theoretical properties and practical advantages, the energy loss is adopted as the empirical training objective in this study.

2.2 Engression: A Pre-Additive Modeling Approach

Introduced by Shen and Meinshausen (2024), the Engression framework integrates distributional regression with a pre-additive noise strategy to address input-level uncertainty, offering a new perspective in the literature. Consistent with the motivation outlined in

Chapter 1, this approach reflects the inherently stochastic nature of inputs and avoids the limitation of providing no information beyond the training support, unlike post-additive models. Consequently, the model can now access response values associated with inputs that extend beyond the observed domain, facilitating data-driven extrapolation of the conditional distribution.

To formalize this intuition, the Engression model can be framed as a generative approach that defines the conditional distribution $P(Y \mid X)$ through the transformation of a noise variable. Let $\mathcal{M}$ denote a class of functions $g(x, \eta)$ where g is deterministic and $\eta \in \mathbb{R}^{d_\eta}$ is a noise vector drawn independently of X from $\mathcal{D}$. Each function $g \in \mathcal{M}$ defines a conditional distribution $P_g(y \mid x)$ through the transformation $Y = g(x, \eta)$. The goal is to identify a function $\tilde{g} \in \mathcal{M}$ that minimizes the expected discrepancy between $P_g(y \mid x)$ and the true conditional distribution $P(y \mid x)$, measured through a proper scoring rule $\mathcal{L}$. This leads to the following population-level objective

$$\tilde{g} \in \arg\min_{g \in \mathcal{M}} \mathbb{E}_{X \sim P_X} \mathbb{E}_{Y \sim P(y|X)} \left[\mathcal{L}(P_g(y \mid X), Y) \right], \tag{4}$$

where $\mathcal{L}(\cdot)$ is chosen to be the energy score, as previously defined and justified in Section 2.1.

To train the model on finite data, the population objective is approximated by a sample-based estimator. Let $\{(x_i, y_i)\}_{i=1}^n$ denote a dataset of observations, and for each i, let $\eta_i^{(j)} \sim \mathcal{N}(0, \sigma^2 I_{d_\eta})$ for $j = 1, \ldots, m$ denote m independent noise realizations drawn from a fixed homoscedastic Gaussian distribution. The resulting empirical training energy loss is defined as

$$\hat{\mathcal{L}}(g) := \frac{1}{n} \sum_{i=1}^{n} \left[\frac{1}{m} \sum_{j=1}^{m} \left\| y_i - g\left(x_i, \eta_i^{(j)}\right) \right\| - \frac{1}{2m(m-1)} \sum_{j=1}^{m} \sum_{\substack{k=1 \\ k \neq j}}^{m} \left\| g\left(x_i, \eta_i^{(j)}\right) - g\left(x_i, \eta_i^{(k)}\right) \right\| \right]. \tag{5}$$

Through this framework, we obtain an unbiased estimator of the population loss by training directly on observed input–response pairs with sampled noise by finding

$$\hat{g} \in \arg\min_{g \in \mathcal{M}} \ \hat{\mathcal{L}}(g), \tag{6}$$

where g defines the generative mechanism for approximating the conditional distribution, with $\hat{g}$ learned from the data as its empirical estimate.

Building on this theoretical framework, Engression offers strong theoretical guarantees.

As shown by Shen and Meinshausen (2024, Proposition 1), when the noise distribution is correctly specified and the model class is sufficiently expressive, the learned function $\hat{g}$ converges to the true conditional distribution within the training support as $n \to \infty$. This population-level guarantee reinforces the theoretical foundation of Engression as a valid approach to distributional regression. Beyond this in-support accuracy, Engression additionally exhibits the ability to extrapolate meaningfully beyond the training domain as established in in Shen and Meinshausen (2024, Theorem 1). This property is proved to hold under mild conditions, such as the local monotonicity of g in x for fixed η and bounded support of the noise distribution. Under these conditions, extrapolation stability is guaranteed, and the valid extrapolation region grows proportionally to the extent of the support of η. By contrast, when these assumptions are only partially satisfied, the model reverts to approximately linear behavior at the boundary, offering a robust fallback for handling out-of-distribution inputs. This robustness differs markedly from models such as tree-based methods, which produce constant outputs beyond the training region, and standard neural networks, which often yield unstable predictions due to their sensitivity to initialization.

To implement the formulation in practice, the conditional sampler $g(x, \eta)$ is parameterized using a fully connected neural network. As shown in Figure 1, the input vector x_i is first concatenated with a noise vector $\eta_i^{(j)} \sim \mathcal{N}(0, \sigma^2 I_{d_\eta})$ sampled independently across observations i and noise indices j. Then, the resulting concatenated vector is passed through a multilayer perceptron (MLP), where each dense layer applies a linear transformation followed by a ReLU nonlinear activation function. Lastly, the final hidden representation is mapped to a univariate output through a linear layer, producing a sample $\hat{y}_i^{(j)} := \hat{g}\big(x_i, \eta_i^{(j)}\big)$ drawn from the modeled conditional distribution. In order to increase the expressiveness of the model, multiple hidden layers may be stacked, with each layer feeding its output into the next.

In terms of training, standard deep learning routines are employed, with small adaptations to reflect Engression's generative and distributional structure. For each training input x_i, the network generates m predictions $\hat{y}_i^{(1)}, \ldots, \hat{y}_i^{(m)}$ by sampling independent noise vectors $\eta_i^{(1)}, \ldots, \eta_i^{(m)}$ and performing a forward pass using the current model parameters. These samples are then compared with the observed response y_i via the empirical energy loss (5), which quantifies the distributional quality of the predictions. Based on this loss, gradients are computed through backpropagation and used to update the network parameters using the Adam (Kingma and Ba, 2014) optimizer, which is known for its stability and adaptive per-parameter learning rates. By repeating this process over multiple epochs, the model progressively improves its internal representation, leading the learned function $\hat{g}$ towards

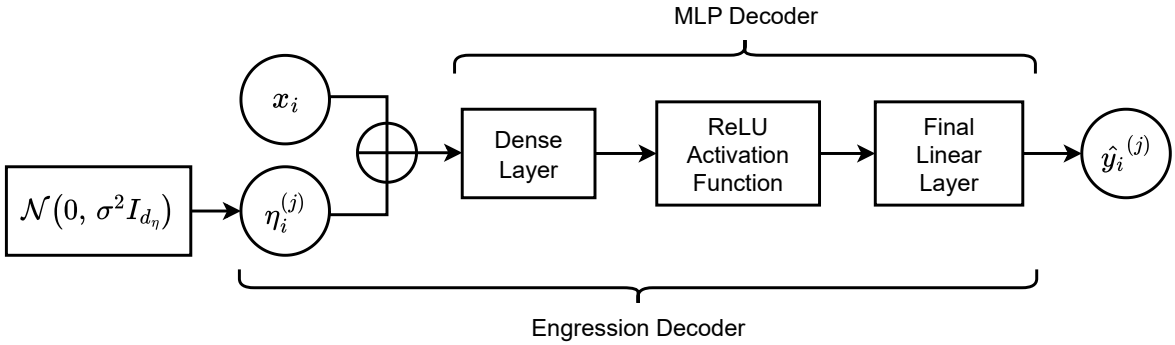

Figure 1: Diagram of the Engression architecture, where the Engression decoder concatenates ($\oplus$) covariates with an independent noise vector before being processed through a multilayer perceptron (MLP).

an accurate approximation of the conditional distribution $P(Y \mid X)$.

Through its architectural design and training methodology, Engression provides a straightforward mechanism for estimating key characteristics of the conditional distribution, such as the mean and quantiles. Once the model has been sufficiently trained, conditional samples at a given input x_i can be obtained by drawing $\eta_i^{(j)} \sim \mathcal{N}(0,\, \sigma^2 I_{d_\eta})$, $j = 1, \ldots, m$, and evaluating $\hat{g}(x_i, \eta_i)$ for each draw. The empirical estimators for the conditional mean and α-quantile of Y given $X = x_i$ are computed as follows from the samples generated by the model

$$\hat{\mu}(x_i) = \frac{1}{m} \sum_{j=1}^{m} \hat{g}(x_i, \eta_i^{(j)}) \qquad \hat{q}_\alpha(x_i) = Q_\alpha \left(\left\{ \hat{g}(x_i, \eta_i^{(j)}) \right\}_{j=1}^{m} \right), \tag{7}$$

where Q_α denotes the empirical α-quantile function of the generated values.

Collectively, the architecture, training procedure, and sampling-based inference described above position Engression as a flexible and theoretically grounded framework for distributional regression that addresses the common limitations found in the probabilistic forecasting literature.

2.3 Recurrent Neural Networks for Time Series

With the rise of deep learning in the statistical literature, there has been growing interest in extending neural architectures to sequential data, particularly in applications that require modeling temporal dependencies. However, earlier frameworks such as feedforward neural networks were inefficient for sequential data, as they inherently lacked the capacity to represent evolving temporal context due to their underlying assumption about input

independence. To address this limitation, initial approaches relied on feature engineering techniques that incorporated lagged covariates as static inputs to encode temporal patterns. Although these strategies provided an efficient solution for short-term effects, in practice they often fell short of capturing more intricate dynamics over longer horizons, thus limiting their applicability.

An initial solution was proposed by Elman (1990), with the Recurrent Neural Network (RNN) as a novel architecture specifically designed to handle sequential data. The key idea in this approach lies in the recurrent structure of the neural network, which enables the model to capture temporal dependencies. Unlike conventional methods that treat inputs as independent features in a forward architecture, the RNN uses a single cell that is applied iteratively across the sequence, introducing sequential processing of inputs. Through this simple and effective design, the model is able to establish a feedback loop in which information is passed through a hidden state that evolves over time, serving as dynamic memory.

More formally, as illustrated in Figure 2, at each time step the hidden state $h_t \in \mathbb{R}^{d_h}$ is updated based on the current input $x_t \in \mathbb{R}^{d_x}$ and the previous hidden state $h_{t-1} \in \mathbb{R}^{d_h}$, enabling the model to maintain a dynamic internal representation of the past and use it to predict future steps. The state update is then computed using the following formula, which defines the cell's computational logic

$$h_t = \tanh(Wx_t + Uh_{t-1} + b), \tag{8}$$

where $W \in \mathbb{R}^{d_h \times d_x}$, $U \in \mathbb{R}^{d_h \times d_h}$, and $b \in \mathbb{R}^{d_h}$ are trainable parameters shared across time. Following common practice, the nonlinear activation function $\tanh(\cdot)$ is chosen and applied to ensure a bounded update of the hidden state while allowing the network to capture nonlinear temporal patterns in the data.

To accommodate this recurrent structure, the training approach is also adjusted by employing Backpropagation Through Time (BPTT) instead of standard backpropagation. The procedure relies on unfolding the network over time and sequentially computing parameter updates by aggregating gradients across the sequence. While this approach naturally reflects the architecture's sequential nature, it tends to suffer from the vanishing gradient problem, especially in long sequences. This occurs due to the diminishing magnitude of gradients across distant time steps, which arises from the repeated application of the chain rule over the sequence. The underlying fixed structure of the cell further amplifies the issue, as the memory update passes through the same pathway across time, creating a natural bottleneck. As a result, the model's ability to retain information from

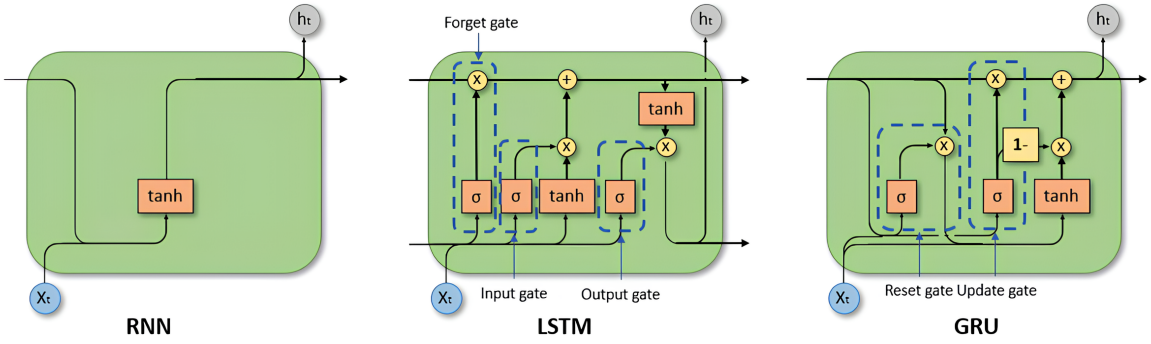

Figure 2: Comparison of RNN, LSTM, and GRU architectures. Source: Dancker (2022).

distant past observations diminishes, limiting its ability to capture long-range temporal dependencies.

In response to the limitations of classical RNNs, adaptations of the original architecture were developed to better capture long-term dependencies in complex domains such as language modeling and time series forecasting. Among these, the Long Short-Term Memory (LSTM, Hochreiter and Schmidhuber, 1997) and the Gated Recurrent Unit (GRU, Cho et al., 2014) gained the most attention, offering a solid alternative for sequential modeling. At the core of both models lies a gating mechanism that adaptively controls memory, replacing the static update rule used in classical RNNs.

In order to improve the flexibility of the memory processing, LSTM introduces an additional internal state responsible for capturing long-term dependencies. To handle this dual memory structure, the architecture is expanded by incorporating three gating mechanisms, as illustrated in Figure 2, which determine how current inputs and past representations are integrated and updated within the cell. While this design has proven effective in several domains, including flood discharge forecasting (Le et al., 2019), some limitations have been observed in practice. Among them, a commonly observed downside lies directly in the architectural complexity, which can increase computational cost and may lead to overfitting, especially when training data are limited.

Developed as a simplified alternative to LSTMs, the GRU model approaches the problem with a more compact architectural design. Rather than maintaining two separate memory flows, it merges the memory cell and hidden state into a single unified representation, mirroring the structure used in classical RNNs. To better regulate this memory structure throughout the sequence, two gates are used, as illustrated in Figure 2. More specifically, the reset gate controls how much of the previous hidden state contributes when combining past and current information, shaping the update of the hidden representation. The update gate then determines how much of this updated state is carried forward, allowing

the model to balance memory retention with the integration of new input. Through this simple architectural design, the model is able to preserve the benefits of gated memory while reducing both computational and structural complexity, offering a practical trade-off between generalization and model flexibility. Although performance remains highly dependent on the nature of the temporal data, GRUs tend to slightly outperform LSTMs on short to medium-length sequences, as shown in many comparative studies (Furizal et al., 2024; Yunita et al., 2025), particularly in settings where consistency and generalization are prioritized over long-range memory capacity.

Despite these architectural advances, gated recurrent models share a structural limitation that becomes evident when modeling time series with specific lag interactions. Due to their underlying design, these models are built to apply the same transformation at each time step recursively, compressing past observations into a single evolving hidden state that serves as the model's memory. While this structure effectively captures temporal continuity and long-range dependencies, it indirectly limits the model's ability to isolate individual lag contributions, as temporal information is bottlenecked within a final representation. A contrasting view is observed in feedforward networks equipped with lagged features that, despite being unable to represent temporal order, can explicitly model each delay by treating lags as independent covariates, thereby recovering positional lag effects without sequence modeling. This contrast reveals a fundamental trade-off in time series forecasting that is sometimes overlooked during model selection, where the choice between sequential and static architectures should reflect the underlying temporal structure and the specific dynamics that need to be captured. To address the limited lag sensitivity of sequential models, attention mechanisms (Bahdanau et al., 2015) have been introduced to allow networks to focus selectively on relevant past time steps rather than rely solely on the final hidden state. These mechanisms, whether static or dynamically learned, are applied across the unfolded hidden states, reintroducing explicit positional information into the representation. Building on this principle, architectures such as the Transformer (Vaswani et al., 2017) have emerged, replacing recurrence entirely with self-attention and achieving state-of-the-art results in some large-scale sequence modeling tasks.

3 Methodology

3.1 Problem Formulation

Building on the theoretical foundations established in Chapter 2, we now formalize the time series forecasting task considered in this work. The goal is to extend the Engression framework to sequential settings in which predictions of the full conditional distribution depend exclusively on past observations, producing a one-step-ahead forecast. While many dynamic regression models, such as ARIMAX (Box and Tiao, 1975), incorporate future covariates at the forecast horizon, either as forecasted covariates or as known future information, our approach relies solely on historical information. This choice establishes a framework aligned with practical applications such as real-time forecasting or next-step alarm systems, where extrapolation capability is valuable to anticipate extreme events.

More formally, given $x_t \in \mathbb{R}^{d_x}$ denoting the vector of covariates observed at time t, and $y_t \in \mathbb{R}$ the corresponding univariate response variable, we define the input space of the s-lag history at time t as

$$\tilde{x}_t = \{(x_\tau, y_\tau)\}_{\tau=t-s}^{t-1} \tag{9}$$

where the lag length s is chosen to capture all relevant temporal dependencies under a simplifying finite-order Markov assumption. In practice, the choice is commonly set according to the forecasting context and observed autocorrelation structure.

Given this definition, the forecasting objective at each step t is to predict the conditional distribution $P(Y_t \mid \tilde{x}_t)$ of the next-step response given the most recent s observations. In line with the Engression formulation introduced in Section 2.2, this task reduces to learning a function $g \in \mathcal{M}$ by minimizing the empirical energy loss defined in Equation (5). The learned approximation $\hat{g}$ is implemented via neural networks as detailed in the following sections.

3.2 Data Preparation

Following the problem formulation defined in Section 3.1, this section presents the data preparation steps applied to both synthetic and real-world datasets used in this study. Within this framework, the intent is to establish a rigorous preprocessing pipeline that respects the temporal structure of sequential data while ensuring fair comparison across all models evaluated. Although both applications involve time series data, the procedures

described can be extended to any sequential dataset where preserving order structure and underlying dependencies is required.

As is common practice in preprocessing, data cleaning represents a critical first step that must be performed carefully according to the nature of the data. In non-sequential datasets, where observations are considered independent, cleaning procedures are typically less constrained. Among these, interpolation and straightforward removal of problematic records are both viable approaches without risk of disrupting underlying patterns. Sequential data, however, demand greater attention to preserve the temporal structure on which these models depend. In this context, temporal interpolation is commonly used to address missing observations while ensuring temporal continuity in the sequence. By contrast, when periods are heavily incomplete, it is often just preferable to remove the entire interval to prevent the model from learning patterns distorted by synthetic interpolated values.

Once the data are cleaned, the dataset is divided into three parts, commonly known as the training set, the validation set, and the test set. As detailed in Section 3.5, the training set will be employed to fit the model parameters, while the validation set will be used for hyperparameter tuning to determine the final configuration under optimization. The test set, by contrast, will be exclusively reserved for evaluating performance, following the methodology outlined in Section 3.6, to assess how well the trained model generalizes on unseen data. In performing these splits, unlike non-sequential settings, where shuffling is commonly used to make subsets representative of the whole dataset, sequential data need to retain their chronological order to preserve temporal dependencies. For this reason, the split is typically based on fixed proportions of the available time horizon, with older observations assigned to the training and validation sets and more recent observations for testing.

As the next step in our preprocessing pipeline, feature-wise standardization is applied to both input and response variables. This step aims to improve training stability by placing all features on comparable scales, which facilitates consistent parameter tuning and stable optimization. Additionally, due to the nature of our framework, standardization ensures that the pre-additive noise perturbs all features equally, preventing scale-dependent distortions when concatenated with the input space.

To ensure correct application, feature-wise standardization is computed on the training set only, with inputs and responses standardized separately to facilitate consistent application. These scaling parameters, computed on the training set, are then applied to the validation and test sets to match the input scale used during training, allowing the optimized model to operate correctly. Through this sequential approach, data leakage is prevented in the

training–evaluation framework by ensuring that no information from the validation and test sets influences the model prior to its evaluation. Finally, an inverse transformation using the learned scaling parameters is applied to both predictions and responses, returning the performance metrics to the original scale of the observed data.

For the univariate response case considered in this work, the standardization transformation is defined as

$$X' = \frac{X - \hat{\mu}_X}{\hat{\sigma}_X}, \qquad y' = \frac{y - \hat{\mu}_y}{\hat{\sigma}_y}, \tag{10}$$

where $\hat{\mu}_X \in \mathbb{R}^{d_x}$ and $\hat{\sigma}_X \in \mathbb{R}^{d_x}$ denote the feature-wise means and standard deviations estimated from X_{train}, and $\hat{\mu}_y, \hat{\sigma}_y \in \mathbb{R}$ are the mean and standard deviation estimated from y_{train}. This transformation is applied to all three sets (train, validation, test) using the parameters computed only from the training set. The corresponding inverse transformation can be derived directly to recover the original scale.

To conclude the preprocessing setup, the temporal input spaces $\tilde{x}_t$ as defined in Equation (9) are constructed according to the requirements of each architecture. For non-sequential Engression models, lagged variables are generated over a fixed history length s and flattened into a feature vector of dimension $s(d_x + 1)$. For sequential architectures, conversely, the same variables are arranged in a tensor of shape $s \times (d_x + 1)$, thereby preserving the temporal order through the introduction of an additional dimension. Incomplete sequences resulting from the shifting process are then discarded, and this data loss is accounted for in the standardization by excluding the first s observations from the training set.

With this constructed pipeline, the original data are now structured into suitable temporal input spaces and prepared for use within the model architectures introduced in the next sections.

3.3 Architectural Extensions

In this section, we present two architectural extensions that adapt the Engression framework to the specific requirements of time series forecasting. Building on the baseline model described in Section 2.2, these modifications are designed to overcome the key limitations of the original formulation when applied to sequential data. The first extension integrates a temporal encoder that processes the s most recent observations by capturing the sequential dependencies among them. The second, by contrast, replaces the fixed noise injection mechanism with an input-dependent formulation, in which the variance of

the injected noise is learned as a function of the input space.

3.3.1 Temporal Encoding

Drawing from the theoretical foundations presented in Section 2.3, this subsection introduces the first extension of the original model through the integration of a temporal encoder. The objective is to learn a latent representation $\tilde{h}_t \in \mathbb{R}^{d_h}$ that captures the temporal dependencies of the input space by moving beyond the static treatment of lagged covariates imposed by the baseline formulation.

Among the recurrent architectures evaluated, GRU was selected as the temporal encoder for this extension. This choice was initially motivated by the theoretical advantages that GRU demonstrates over LSTM in regression tasks, as discussed in Section 2.3. To further support this decision, additional synthetic experiments were conducted to compare their performance as encoders in distributional regression. These results showed that LSTM often tended to overfit the training data during encoding, leaving limited capacity for the Engression decoder to model the conditional distribution. By contrast, GRU's simpler architecture produced lower-complexity representations that preserved sufficient flexibility for the Engression decoder to exploit the extrapolation benefits of the pre-additive formulation. Given these properties and the focus on short to medium-length sequences in this work, the GRU encoder was adopted for all subsequent experiments. For completeness, additional evaluations with RNN and pre-trained Transformer models were performed, but both were ultimately discarded due to representational limitations in the former case and excessive computational cost relative to the problem scale in the latter.

More formally, as illustrated in Figure 3, the input space $\tilde{x}_t \in \mathbb{R}^{s\times(d_x+1)}$ is sequentially passed through the shared recurrent cell of the GRU encoder. During this process, at each time step τ within the historical window, an updated hidden state $h_\tau \in \mathbb{R}^{d_h}$ is iteratively computed, providing a latent representation of the information accumulated in the sequence. The resulting collection of hidden states is given by

$$H_t = \{h_{t-s+1}, \ldots, h_t\}, \tag{11}$$

where s denotes the length of the historical window considered in the modeling.

Once H_t has been computed by the recurrent encoder, standard practice relies on using the final hidden state h_t as a summary representation of the entire sequence. While conceptually correct, this practice often creates a bottleneck, particularly in longer sequences, where the entire history must be compressed into a fixed-size representation,

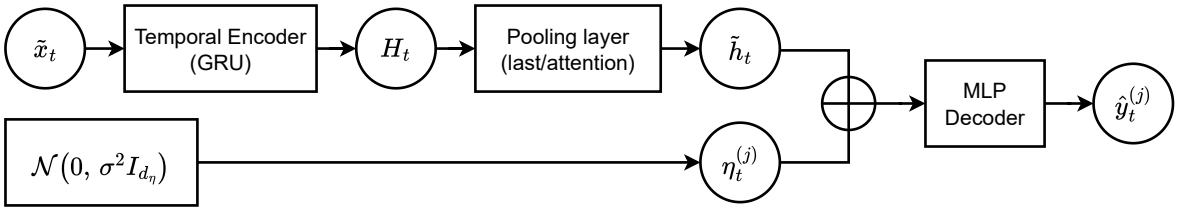

Figure 3: Diagram of the Sequential Engression architecture, extending Engression with a pooled temporal encoder of the input space.

thereby limiting the information quality available to the Engression decoder. Beyond this limitation, as highlighted in Section 2.3, the shared recurrent structure additionally hinders the model from explicitly capturing individual lag contributions, reducing its ability to capture lag-specific dependencies. This issue becomes particularly evident in forecasting autoregressive processes (Box et al., 2015) commonly encountered in time series tasks, where individual past observations carry distinct predictive importance compared to the sequential accumulation of temporal information.

To address these limitations, several pooling and attention mechanisms were explored as alternatives to the last hidden state. Among them, mean and max pooling were initially tested as summaries of the sequence history H_t, but they proved to be ineffective likely due to their oversimplified treatment of lag effects. As a more complex alternative, dynamic attention mechanisms were investigated, assigning input-dependent attention weights to the hidden states contained in H_t. However, while more flexible than previous alternatives, they introduced computational cost and training instability, making them unsuitable for the task. As a compromise, a static attention mechanism was employed to offer a more robust alternative by assigning fixed learnable weights to each hidden state of the sequence independently of the input space.

More precisely, the architecture relies on constructing a weighted sum of the hidden states in H_t, with softmax normalization to further ensure stability and proper weight constraints. The resulting pooled representation $\tilde{h}_t$ is then defined as

$$\tilde{h}_t = \sum_{i=1}^{s} \alpha_i \, h_{t-s+i}, \qquad \alpha_i = \frac{\exp(w_i)}{\sum_{k=1}^{s} \exp(w_k)}, \tag{12}$$

where the attention weights $\alpha_i \in \mathbb{R}$ capture the relative importance of each lag in the historical sequence, with scores parameterized by learnable scalars $w_i \in \mathbb{R}$.

As a result, the latent representation $\tilde{h}_t$ is defined either as the final hidden state h_t or as the pooled representation computed through the static attention mechanism, depending

on the chosen pooling strategy. This representation is then passed to the Engression decoder, maintaining the original pre-additive framework while enabling the model to process temporal information.

3.3.2 Input-Dependent Noise Injection

Building on the sequential architecture introduced above, this subsection presents the second extension of Engression via the introduction of heteroskedastic noise injection. Through this extension, we aim to address the key limitation of the original framework, which relies on static pre-additive noise sampled from a homoscedastic distribution. Although this noise assumption may commonly hold in static settings, it often fails to capture real-world time series where conditional variance exhibits time-varying patterns such as volatility clustering. As a result, this lack of flexibility restricts the model's ability to produce accurate conditional distributions that reflect the true uncertainty structure of the process. To overcome this constraint, the objective is to learn an input-dependent covariance matrix $\Sigma(\tilde{h}_t) \in \mathbb{R}^{d_\eta \times d_\eta}$ that characterizes the pre-additive noise distribution, enabling the model to capture heteroskedastic dynamics.

Formally, the objective is to model dynamic pre-additive noise, $\eta_t \sim \mathcal{N}(0,\, \Sigma_t(\tilde{h}_t))$, where the covariance matrix $\Sigma_t(\tilde{h}_t)$ is parameterized as a function of the conditional variance $\tilde{\sigma}_t^2(\tilde{h}_t)$. To account for the varying complexity of heteroskedastic dynamics, two parameterizations are considered in this architecture, formulated as

$$\begin{aligned} &\text{Scalar representation:} && \Sigma_t(\tilde{h}_t) = \tilde{\sigma}_t^2(\tilde{h}_t)\, I_{d_\eta}, && \tilde{\sigma}_t^2(\tilde{h}_t) \in \mathbb{R}; \\ &\text{Vectorized representation:} && \Sigma_t(\tilde{h}_t) = \operatorname{diag}\Big(\tilde{\sigma}_t^2(\tilde{h}_t)\Big), && \tilde{\sigma}_t^2(\tilde{h}_t) \in \mathbb{R}^{d_\eta}. \end{aligned} \tag{13}$$

In the scalar formulation, a single variance parameter is shared across all noise dimensions, extending the original framework with an input-dependent noise level. By contrast, the vectorized formulation increases the flexibility of the model by learning distinct variance terms for each noise dimension, thereby enabling richer heteroskedastic modeling.

In the proposed implementation, the conditional variance $\tilde{\sigma}_t^2(\tilde{h}_t)$ and the resulting covariance matrix $\Sigma_t(\tilde{h}_t)$ are modeled via a dedicated conditional noise encoder, as illustrated in Figure 4. More specifically, the encoder applies a linear projection of the latent representation $\tilde{h}_t$, followed by the Softplus activation function (Dugas et al., 2001). Due to the shared nature of the modeling objective across both parameterizations, the encoder maintains the same architectural structure, differing only in the dimensionality of the learned output, taking the form

$$\begin{aligned} &\text{Scalar representation:} && \tilde{\sigma}_t^2(\tilde{h}_t) = \text{softplus}(w^\top \tilde{h}_t + b), \quad w \in \mathbb{R}^{d_h},\ b \in \mathbb{R}; \\ &\text{Vectorized representation:} && \tilde{\sigma}_t^2(\tilde{h}_t) = \text{softplus}(W\tilde{h}_t + b), \quad W \in \mathbb{R}^{d_\eta \times d_h},\ b \in \mathbb{R}^{d_\eta}, \end{aligned} \tag{14}$$

where w, W, and b denote the learnable parameters optimized during training.

In developing the proposed encoder, both theoretical principles and empirical testing were taken into account for the two key design choices just formalized. In evaluating network depth, the architecture was initially expanded into a feedforward neural network but was later discarded due to higher computational cost and only marginal improvement in predictive quality. This result could be attributed to the pooled temporal encoder already embedding nonlinearity and complexity in the latent representation $\tilde{h}_t$, making deeper projections redundant in the modeling of $\tilde{\sigma}_t^2(\tilde{h}_t)$. For this reason, a simple linear projection was set as a suitable trade-off, retaining flexibility without introducing additional model complexity and training instability, particularly in the vectorized formulation where the number of learnable parameters already increases substantially. Separately, the choice of activation function stems directly from the nature of the variable to be modeled. Given the strictly positive nature of the variances, a suitable smooth mapping to $(0, \infty)$ was required. Among the options tested, Softplus, defined as

$$\text{Softplus}(x) = \log(1 + e^x), \tag{15}$$

was preferred over both the ReLU activation and the common log-variance parameterization often employed in probabilistic models. This preference arises from its strictly positive nature over the whole domain, which keeps variance parameters positive, even for small or negative pre-activation values, thereby preserving the stochastic nature of the model. Additionally, Softplus serves as a smooth approximation of ReLU, providing more stable gradients during optimization, as demonstrated by Glorot et al. (2011) and confirmed in early testing. Lastly, faster convergence was observed with Softplus compared to both ReLU and log-variance parameterization, providing the final validation for this architectural choice.

As a result of the proposed framework, each forward step generates a latent representation $\tilde{h}_t$ and a corresponding conditional noise distribution $\mathcal{N}(0,\, \Sigma_t(\tilde{h}_t))$ from which a noise vector $\eta_t^{(j)}$ is sampled. This noise vector is then concatenated with the latent representation according to the original Engression framework producing a sample $\hat{y}_t^{(j)}$ drawn from the modeled conditional distribution. Through the integration of this dual-encoder structure,

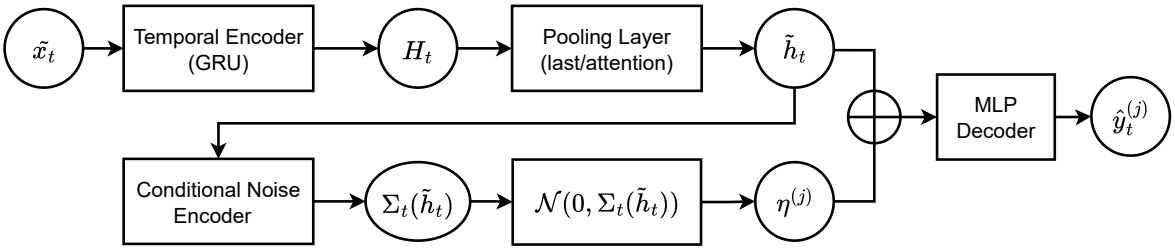

Figure 4: Diagram of the Heteroskedastic Sequential Engression architecture, extending Sequential Engression with input-dependent noise injection.

the model extends the original Engression beyond its static limitations by enabling the learning of both sequential dependencies and heteroskedastic dynamics.

3.4 Model Configurations

To assess the effectiveness of the proposed extensions, we conduct a systematic comparison with the Engression baseline and corresponding deterministic counterparts. Through this framework, we aim to isolate the marginal contributions of each component in the architecture by evaluating their impact on time series forecasting. The models considered are as follows:

- **MLP**: Serves as the deterministic counterpart of Engression, implementing a feed-forward neural network with the same flattened input space. Due to the absence of noise injection, the resulting model is trained to estimate the conditional mean $\mathbb{E}(Y_t \mid \tilde{x}_t)$, providing a point forecasting baseline for comparison.

- **Sequential MLP**: Serves as the deterministic counterpart of Sequential Engression by using the same pooled temporal encoder but directly applying a standard MLP decoder to the latent representation $\tilde{h}_t$ without noise injection. Following the same approach as the MLP, this model estimates the conditional mean $\mathbb{E}(Y_t \mid \tilde{x}_t)$, providing a deterministic comparison for the sequential architecture.

- **Engression**: Implements the original framework proposed by Shen and Meinshausen (2024), relying on the homoscedastic pre-additive noise structure. Given the lack of temporal encoding, the input space $\tilde{x}_t$ is constructed as a flattened vector of lagged features, following the procedure outlined in Section 3.2.

- **Heteroskedastic Engression**: Builds on the heteroskedastic sequential extension introduced in Subsection 3.3.2 by omitting the pooled temporal encoder. As a

result, the flattened input space $\tilde{x}_t$ assumes the dual role of $\tilde{h}_t$ as both input to the Engression decoder and argument to the conditional noise encoder, while maintaining the rest of the internal architecture unchanged.

- **Sequential Engression**: Extends the original model with a temporal encoder to process the sequential input space $\tilde{x}_t$, following the architecture detailed in Subsection 3.3.1.
- **Heteroskedastic Sequential Engression**: Combines the temporal encoder with input-dependent noise injection, as described in Subsection 3.3.2, representing the full extension of Engression for time series forecasting.

For completeness, diagrams of the supplementary architectures (MLP, Sequential MLP, and Heteroskedastic Engression) are provided in Appendix A.1, while Figures 3 and 4 illustrate the discussed extensions (Sequential Engression and Heteroskedastic Sequential Engression).

3.5 Training and Validation Framework

Following the definition of each model architecture, this section introduces the procedure adopted to ensure consistent identification of optimal configurations for each model employed. Among the validation strategies considered, temporal cross-validation with rolling windows (Tashman, 2000) was initially implemented but ultimately discarded. This decision stemmed from the limitations imposed by the sequential nature of time series data, which restricts the reuse of observations across folds. This constraint introduces a critical trade-off between the benefit of improved generalization during hyperparameter tuning and the drawback of reduced training data within each fold. With the increase in folds, the amount of data available for training becomes progressively smaller, leading validation procedures to favor simpler architectures that generalize better on smaller training subsets. Consequently, the hyperparameter configurations identified may not reflect those optimal when trained on the larger set, making this approach unreliable in settings with limited data and distributional models that require extensive training samples. To address the inefficient reuse of data, an expanding-window strategy (Tashman, 2000) was also tested but rejected due to varying training data size across folds, which compromises the generalization of hyperparameters. As a result, we adopted an alternative approach that employs a simpler chronological train-validation split, prioritizing broader exploration of the hyperparameter space at the cost of higher variance in model selection.

Within this framework, the chosen procedure follows standard practices for model optimization. For each hyperparameter configuration, we train the corresponding model on the designated training set and evaluate its predictive performance on the validation set. This process is repeated across all candidate hyperparameter combinations. The resulting best-performing configuration in terms of validation loss is then retained for final model evaluation on the test set.

The training procedure for each configuration begins with random initialization of model parameters, following standard practices adapted to the activation functions and layer types employed (Goodfellow et al., 2016). At each training iteration, a forward pass computes predictions that are evaluated using the respective loss function, with energy loss applied to Engression-based models and mean squared error to deterministic ones. For Engression-based models specifically, two forward samples per input are used to balance computational efficiency with gradient stability in energy loss estimation, as recommended in the original work (Shen and Meinshausen, 2024). The resulting loss is then backpropagated through the network, allowing parameter updates via the Adam (Kingma and Ba, 2014) optimizer, which facilitates adaptive learning rates through gradient descent. This cycle of forward propagation, loss computation, and parameter updates is repeated over multiple epochs until convergence, resulting in a final optimized set of model parameters for the chosen hyperparameters.

To determine the optimal number of training epochs without extensive grid search, we employed the early stopping strategy (Prechelt, 1998) across all model configurations. At its core, during every epoch, the validation loss is computed and monitored by evaluating the generalization performance of the current model on the validation set. This strategy allows the model to stop training when the validation loss fails to improve by more than a fixed threshold over a patience period of consecutive epochs. The use of a patience criterion is driven by the inherently noisy nature of sample-based loss estimation, making early stopping more robust to local fluctuations and suboptimal local minima in the validation loss. As a result, the model avoids both underfitting from insufficient training and overfitting from excessive optimization by identifying the optimal configuration for final evaluation on the test set. Compared to common practice, a minor adjustment is introduced to reduce the computational overhead of the early stopping procedure. For both deterministic and Engression-based models, the validation loss is computed every other epoch rather than at every iteration. While this approach may slightly reduce the precision in identifying the optimal stopping point, preliminary tests on synthetic data suggested that the computational savings justified the trade-off, with minimal downside. To further reduce computational cost in the Engression-based models implemented, the

number of forward samples used during validation is reduced to 50, compared to 100 during evaluation, maintaining sufficient precision while reducing validation time.

An additional training challenge arises from the limitations introduced by mini-batch training in the Engression-based models studied in this work. While mini-batch training represents standard practice in deterministic models due to the computational efficiency and faster convergence it provides (Bottou, 2010), these advantages were not observed in our implementations. The issue stems from the additional randomness introduced by mini-batch sampling variability, which, when combined with pre-additive noise, leads to noisy gradient estimates that can harm convergence. To mitigate this issue, full-batch training offers a more stable alternative by eliminating the added sampling variability and producing more accurate parameter updates. In practice, however, this approach becomes increasingly memory-intensive, with even higher computational costs in the proposed sequence modeling architectures, as longer windows inherently expand the input dimensions. In such cases, full-batch training becomes computationally infeasible, forcing constraints on model capacity due to hardware limitations. To address this constraint, we adopted a practical compromise by using the largest batch size supported by available computational resources for each experiment, maximizing the stability of the training procedure while ensuring computational feasibility.

Beyond these training procedures, regularization techniques have been explored to improve the generalization capabilities of the proposed architectures. Among standard regularization practices, weight decay (Krogh and Hertz, 1991) was chosen over the widely adopted dropout (Srivastava et al., 2014). This choice reflects both theoretical and practical considerations observed during preliminary testing. From a theoretical perspective, as highlighted by Helmbold and Long (2017), dropout tends to impose an exponential penalty on large activations, whereas weight decay applies a polynomial penalty that preserves relative activation differences while controlling the overall scale of model parameters. This distinction is particularly relevant for modeling tasks that must accurately capture tail behavior, as dropout would suppress activation signals from extreme events of interest, limiting the model's learning and prediction capabilities. Additionally, early testing revealed that dropout frequently introduced training instabilities due to the negative interaction between its randomness and the pre-additive noise mechanism already present in these types of models. By contrast, weight decay preserved training stability through its deterministic application of regularization while effectively controlling overfitting. Lastly, at a practical level, weight decay uniformly applies penalties across all layers, avoiding the need to tune layer-specific dropout rates that would significantly increase the hyperparameter exploration burden already imposed by these complex architectures. These combined

considerations have led to the consistent adoption of weight decay regularization across both deterministic and Engression-based model variants.

Finally, in parallel with these training mechanics and regularization choices, a unified grid search strategy is adopted for both experiments to ensure a fair and consistent comparison across all models. To this end, the learning rate and weight decay are tuned over the same candidate values for every model, and identical architectural configurations are evaluated across each extension and its corresponding deterministic counterpart. For the Engression-based models, the noise dimension d_η is fixed to one hundred, and the noise distribution is set to a standard normal distribution $\mathcal{N}(0, I_{d_\eta})$, following the experimental guidelines of the original paper (Shen and Meinshausen, 2024). A complete list of configurations tested is provided in Appendix A.2, with specific hyperparameter values selected based on established literature practices and preliminary testing during implementation.

3.6 Evaluation Procedure

Building on the training-validation framework established above, this section presents the evaluation metrics computed on the test set using the optimal model configurations identified. The aim is to provide a comprehensive evaluation methodology that enables a deeper understanding of the effectiveness of the proposed extensions while establishing a comparison framework with the corresponding deterministic models. To account for the differing nature of synthetic and real-world datasets, two sets of metrics are employed depending on whether the true conditional distributions are known, allowing for more precise evaluation in the simulation setting. Furthermore, in line with the focus of this thesis on extreme event forecasting, additional metrics are incorporated to examine the model's predictive capacity in extrapolative regimes. Lastly, to ensure reliable results, each model is trained and evaluated across ten independent runs with different random seeds, capturing variability from both initialization and optimization dynamics. The resulting metrics covered in this evaluation approach are organized into three categories as outlined below.

Shared metrics applied to both deterministic and Engression-based models:

- **MSE**: Mean squared error serves as the standard loss function for deterministic models, where it evaluates the squared deviation between the predicted response $\hat{y}_t$ and the observed value y_t. In the context of Engression, MSE can be estimated by computing the empirical mean $\hat{\mu}(\tilde{x}_t)$ from multiple forward passes as defined in Equation (7). This formulation enables a direct comparison between the two types

of models by assessing how accurately each framework approximates the conditional mean of the target variable. The respective computations are

$$\text{MSE}_{\text{det}} = \frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2\,, \qquad \text{MSE}_{\text{eng}} = \frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{\mu}(\tilde{x}_t))^2\,, \tag{16}$$

where the subscripts det and eng indicate whether the model is deterministic or Engression-based, respectively, and T denotes the length of the test set.

- **Extreme Upper Quantile MSE (Extreme MSE)**: Computes the MSE defined in Equation (16), restricted to test observations whose true responses exceed the 99.5% quantile of the training set responses. This metric provides a shared measure of extrapolation capability across both model types by evaluating performance on extreme patterns rarely observed during training. The threshold level is chosen to ensure sufficient observations for stable evaluation while maintaining minimal model exposure to such extremes during training.

Core Distributional Metrics used across both experimental settings:

- **Energy Loss**: Represents the standard loss function for Engression-based models, evaluated as defined in Equation (5) over the m sampled predictions $\hat{y}_t^{(j)}$ at each input $\tilde{x}_t$. This metric quantifies the overall quality of the predictive distribution by jointly assessing sharpness and calibration with respect to the true response y_t.

- **Coverage (PICP)**: Measures the calibration of forecasted distributions by computing the empirical probability that true observations fall within specified prediction intervals derived from the sampled conditional distribution. In our evaluation framework, we assess the 80% prediction interval, which provides an adequate coverage assessment while remaining computationally feasible given the number of samples m generated during evaluation. The metric is defined as

$$\text{PICP}_{80} = \frac{1}{T}\sum_{t=1}^{T}\mathbf{1}\left(\hat{q}_{0.1}(\tilde{x}_t) \le y_t \le \hat{q}_{0.9}(\tilde{x}_t)\right), \tag{17}$$

where $\mathbf{1}(\cdot)$ is the indicator function and $\hat{q}_{0.1}(\tilde{x}_t)$, $\hat{q}_{0.9}(\tilde{x}_t)$ are the empirical 10% and 90% quantiles computed via the sample-based procedure outlined in Equation (7).

- **Sharpness**: Evaluates the precision of the predictive distribution by measuring the average width of its prediction intervals. By computing it for the same interval used

for coverage, it quantifies how specific the forecast is, complementing the calibration assessment. It is defined as

$$\text{Sharpness}_{80} = \frac{1}{T}\sum_{t=1}^{T}\left(\hat{q}_{0.9}(\tilde{x}_t) - \hat{q}_{0.1}(\tilde{x}_t)\right). \tag{18}$$

Ground Truth Distributional Metrics leveraging known true distributions in simulation settings:

- **Quantile MSE**: Evaluates the squared deviation between the estimated quantile $\hat{q}_\alpha(\tilde{x}_t)$ and the true quantile $q_\alpha(\tilde{x}_t)$, across three quantile levels $\alpha \in \{0.1, 0.5, 0.9\}$. This allows evaluation of predictive accuracy across different regions of the conditional distribution, thereby evaluating the model's capacity to capture both central trends and tail behavior. Empirically, it is computed as

$$\text{Quantile MSE}_\alpha = \frac{1}{T}\sum_{t=1}^{T}\left(q_\alpha(\tilde{x}_t) - \hat{q}_\alpha(\tilde{x}_t)\right)^2. \tag{19}$$

To enhance interpretability, all MSE-based metrics (MSE, Extreme Upper Quantile MSE, and Quantile MSE) are reported as their corresponding RMSE by taking the square root of the respective values. In doing so, all measures are presented on the original scale of the response variable, facilitating clearer interpretation of the practical significance of performance differences across models. Furthermore, for cross-seed comparison, both the mean and standard deviation of each measure across seeds are computed to observe their average performance along with their variability.

3.7 Implementation Details

During the early stages of this work, an attempt to adapt the software implementation of Engression by Shen and Meinshausen (2024) was made but ultimately failed. Behind this failure lies the construction of the package itself that, due to extensive computational optimizations, strongly limits the flexibility needed to implement our methodology under full traceability. As a result, all code was rebuilt from scratch to ensure full implementation control and transparency, prioritizing research flexibility over computational optimization. For this implementation, all models and experiments were implemented in Python, with core architectures developed using the PyTorch library to enable CUDA usage on GPU.

In terms of computational resources, all experiments have been trained and evaluated using a personal workstation supported by Intel Core i7-4930K, NVIDIA GeForce GTX 1060 with 6 GB of video memory, and 16 GB of system memory. The full code including model architecture, pipeline, and experiments is available in a private GitHub repository and can be shared upon request, since the real-world dataset used in the application cannot be redistributed.

4 Simulation Study on a Synthetic Time Series

4.1 Process design

In this section, we evaluate the proposed models using a synthetic time series setup with known ground truth dynamics. The aim is to assess the marginal benefit of each architectural extension in improving the forecasting capability of the original model. Furthermore, as outlined in Section 3.6, this controlled setup allows for a more informative evaluation of distributional forecasts by comparing predicted quantiles against their true counterparts. At its core, the generative process is constructed to reflect three key features addressed by the baseline model and respective extensions:

- **Sequential Dependencies**: Are introduced in both the mean and the variance of the process through a latent state variable and lagged relationships of the response and covariates. This design choice aims to evaluate the capacity of the temporal encoder to better capture underlying sequential patterns compared to the lagged independent features implied by the original structure. The latent state accumulates information from recent past responses and covariates, including their interactions, creating a sequential memory effect. By contrast, lagged effects of past features are implemented with decreasing weights reflecting the gradual attenuation commonly observed in temporal dynamics.
- **Heteroskedastic Variance**: Is modeled through a time-varying conditional mechanism based on past variance, the latent state, and recent responses and covariates. This structure provides a direct test of the benefit of input-dependent noise injection compared to the homoscedastic pre-additive noise of the original model. To ensure positivity and capture volatility clustering, squared terms of variables and a GARCH-like dependence on past variance are incorporated. In addition, a constant baseline variance is added to ensure positive variance and maintain a minimum level of uncertainty throughout the process.
- **Pre-additive Noise Structure**: Is implemented by perturbing the conditional mean through the heteroskedastic noise just defined. This design aligns with the original Engression framework by applying a deterministic function over a noise-perturbed signal. For this purpose, a scaled exponential function was chosen as monotonic transformation to preserve the theoretical properties of extrapolation beyond the support discussed in Section 2.2.

Building on these design principles, we now formalize the synthetic data generation process. For a dataset $D = \{(x_t, y_t)\}_{t=1}^{T+T_{\text{burn}}}$ with T effective observations and T_{burn} burn-in samples, each component is sequentially generated from the time series

$$\begin{cases} X_t = \rho_x \, X_{t-1} + \sigma_x \, \varepsilon_t & \varepsilon_t \sim \mathcal{N}(0,1) \\ S_t = \alpha_s \, S_{t-1} + \beta_x \, X_{t-1} + \beta_y \, Y_{t-1} + \gamma_{xy} \, X_{t-1} \, Y_{t-1} \\ \mu_t = \lambda_s \, S_t + \sum_{i=1}^{3} \lambda_{x,i} \, X_{t-i} + \sum_{i=1}^{3} \lambda_{y,i} \, Y_{t-i} \\ \sigma_t^2 = \omega + \phi \, \sigma_{t-1}^2 + \psi_s \, S_t^2 + \sum_{i=1}^{3} \psi_{x,i} \, X_{t-i}^2 + \sum_{i=1}^{3} \psi_{y,i} \, Y_{t-i}^2 \\ U_t = \mu_t + \sigma_t \, \eta_t & \eta_t \sim \mathcal{N}(0,1) \\ Y_t = \exp\left(\tau \, U_t\right) \end{cases} \tag{20}$$

with specific parametrization

Covariate dynamics: $\rho_x = 0.75, \quad \sigma_x = 0.5$

Latent state: $\alpha_s = 0.4, \quad \beta_x = 0.2, \quad \beta_y = 0.2, \quad \gamma_{xy} = 0.1$

Mean coefficients: $\lambda_s = 0.5, \quad \lambda_{x,i} = (0.15, 0.1, 0.05), \quad \lambda_{y,i} = (0.15, 0.1, 0.05)$

Variance coefficients: $\omega = 0.1, \quad \phi = 0.2, \quad \psi_s = 0.2$

$\psi_{x,i} = (0.1, 0.05, 0.01), \quad \psi_{y,i} = (0.1, 0.05, 0.01)$

Other parameters: $\tau = 0.15, \quad T_{\text{burn}} = 1000.$

The choice of each coefficient value results from a trade-off between process complexity and stability, ensuring sufficient dynamics to meaningfully evaluate the architectural extensions while avoiding explosive behavior in the series. Regarding the covariate's dynamics in X_t, a simple autoregressive process was chosen to maintain simplicity while enabling sufficient sequential dependence to be transmitted through the mean and variance processes. In addition, the first $T_{\text{burn}} = 1000$ draws are discarded as burn-in to reduce the influence of initialization and to allow the process to converge to its stationary distribution.

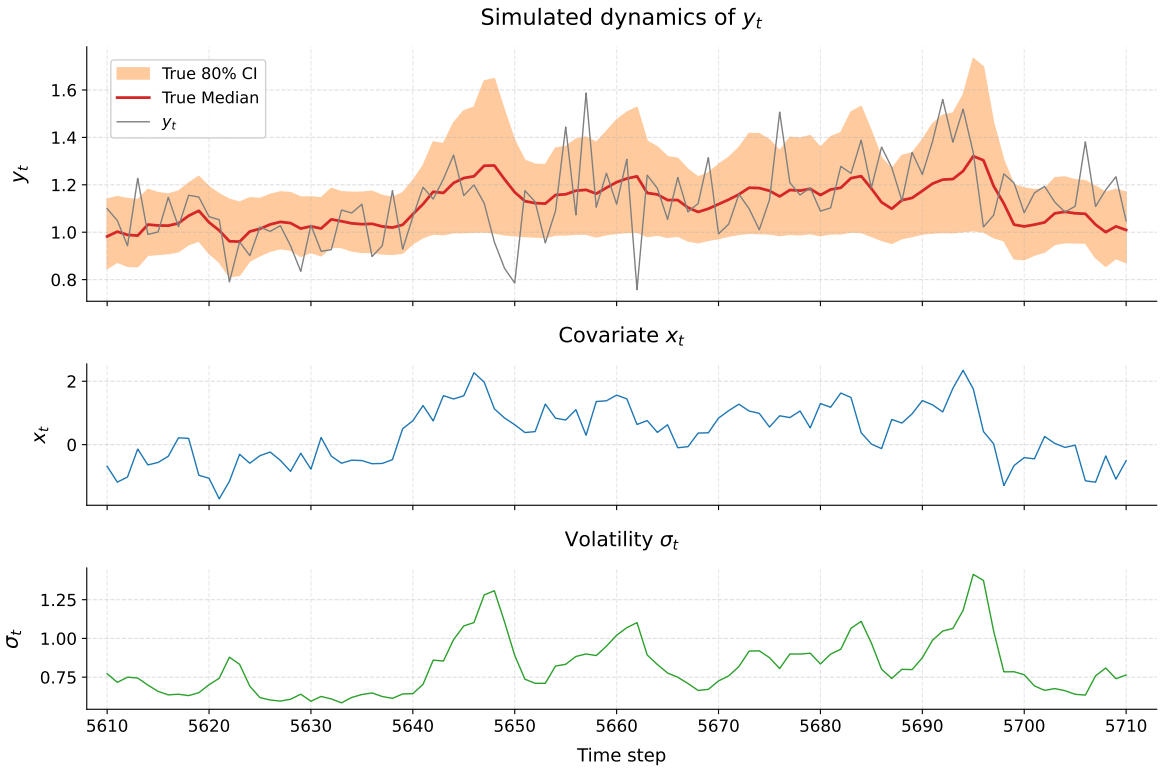

Figure 5: Synthetic time series characteristics over a window of 100 observations.

An important advantage of this formulation lies in that closed-form conditional quantiles are available by exploiting the strictly increasing exponential function chosen. Given that $\log Y_t = \tau U_t \sim \mathcal{N}(\tau\mu_t, (\tau\sigma_t)^2)$, this yields

$$Q_\alpha(Y_t \mid \mu_t, \sigma_t) = \exp\left(\tau\mu_t + \tau\sigma_t\, \Phi^{-1}(\alpha)\right), \tag{21}$$

where $\Phi^{-1}(\alpha)$ denotes the standard normal quantile function for quantile level $\alpha \in (0, 1)$.

To visualize these process characteristics, Figure 5 showcases 100 observations of a simulated series along with the corresponding true quantiles and variance dynamics, demonstrating the sequential structure and heteroskedastic behavior that the proposed models are meant to capture.

4.2 Experimental Setup

Based on the process design just defined, two synthetic datasets of length $T = 7000$ are generated, each obtained with a different random seed. The first dataset is employed for training and validation procedures, while the second serves exclusively for evaluation assessment. By this design, the two datasets can be regarded as independent realizations of the same stochastic process, thereby preventing information leakage in the evaluation

procedure while maintaining comparability across realizations. As illustrated in Figure 6, the first series is divided chronologically, with the first 5000 observations used for training and the remaining 2000 for validation.

To provide sufficient temporal context for one-step-ahead forecast, a history window of ten past observations is employed across all models, yielding input spaces of the form $\tilde{x}_t = \{(x_\tau, y_\tau)\}_{\tau=t-10}^{t-1}$. This choice reflects the autoregressive structure of the data generating process while ensuring a safety margin against misspecification in both mean and variance components.

Following the grid search procedure outlined in Section 3.5, all configurations have been trained on the training set and respectively evaluated over the validation set under the same random seed. For completeness, Appendix A.3.1 provides the optimal configurations along with procedural details regarding the early stopping criterion. With the resulting optimal architectural selection in each model class, the training procedure is repeated across ten random seeds, with final evaluation conducted on the test set to quantify model performance and variability across random initializations.

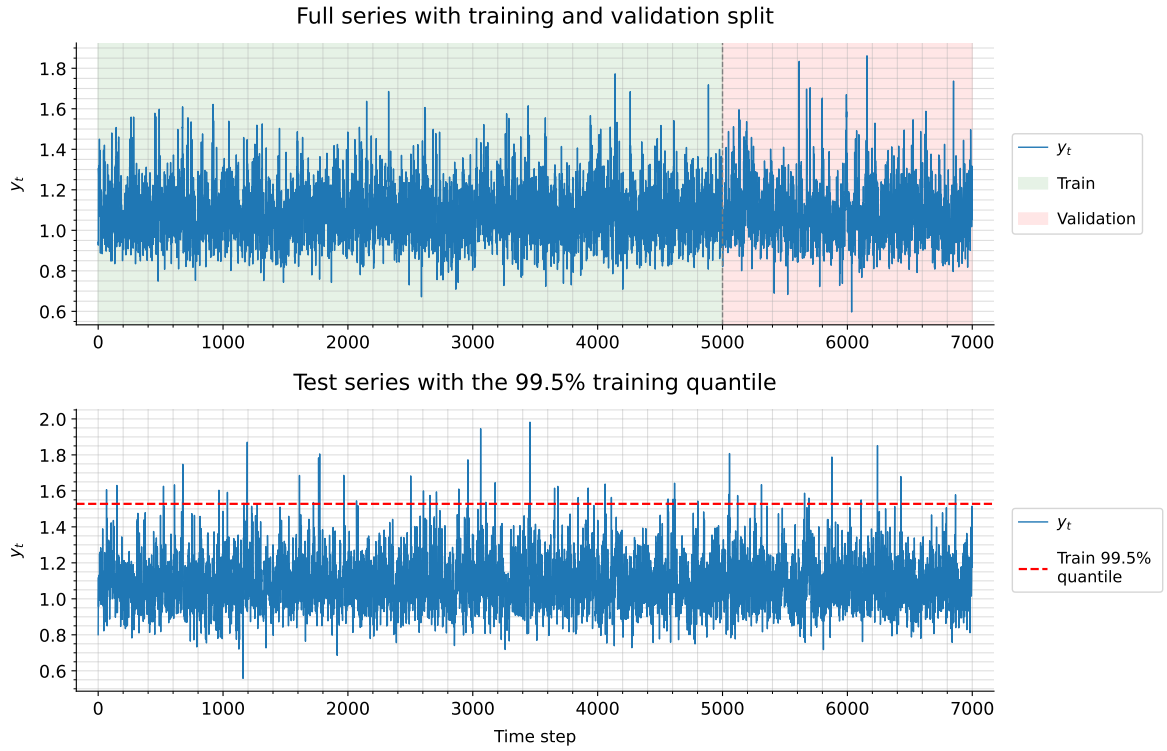

Figure 6: Synthetic series used in the simulation study. The top panel shows the training-validation split, while the bottom panel illustrates the independently generated evaluation series with the 99.5% upper quantile (computed on the training set) highlighted.

4.3 Results

Proceeding to the evaluation results of this simulation case study, several interesting patterns emerge, revealing how architectural complexity affects both performance and stability of the forecast. For reference, all metric scores have been explained in Section 3.6, with cross-seed variability computed across 10 seeds for both plots and tables.

As expected from the simulation's sequential structure, a first confirmation of the benefit introduced by sequential architectures is showcased in Table 1 in mean prediction performances. For deterministic models, Sequential MLP achieves a small marginal RMSE improvement (0.5%) while maintaining comparable extreme prediction performance, with the major benefit being substantial reduction in cross-seed variability. This stability pattern is further highlighted in Figure 7, where standard MLP exhibits much wider variability bands during high-volatility periods, particularly around peaks and troughs of the underlying trend. Regarding Engression-based models, similar small improvements are observed with the Heteroskedastic Sequential Engression demonstrating the strongest performance, achieving 0.4% and 0.7% improvements over baseline Engression in mean and extreme prediction metrics respectively. However, this improvement comes with a trade-off in cross-seed stability, where the baseline Engression model provides much more stable predictions, particularly for Extreme RMSE. Regarding the heteroskedastic extension, minimal marginal gains are observed in mean predictions across model pairs, suggesting this extension offers marginal benefits that may not justify the added complexity. In terms of comparison between model families, deterministic and Engression-based models achieved similar mean prediction performance, with Sequential MLP slightly outperforming Heteroskedastic Sequential Engression by 0.3%. By contrast, Engression-based models demonstrated marginal advantages in extreme event forecasting, with small improvements of 1-1.6% in Extreme RMSE compared to their deterministic counterparts, highlighting their value for extrapolation beyond the training support despite comparable central tendency predictions. For completeness, a visualization of extreme event predictions is provided in Appendix A.3.2, though the noise-driven nature of extremes in this simulation causes systematic underestimation across all architectures, making model comparison difficult on individual observations.

Regarding distributional forecasting quality, metrics of interest are reported in Table 2. In terms of energy loss, marginal improvements of 0.7% are observed in the proposed sequential models, while maintaining similar stability across seeds. Similar gains are additionally observed in the empirical coverage over 80% prediction intervals with Heteroskedastic Sequential Engression reaching the closest to nominal level (78.98%) followed

Model	RMSE	Extreme RMSE	Time (s)
MLP	0.1238 ± 0.0002	0.4379 ± 0.0044	**5.3 ± 0.5**
Seq. MLP	**0.1231 ± 0.0001**	0.4374 ± 0.0018	18.3 ± 3.3
Engression	0.1240 ± 0.0002	0.4334 ± 0.0057	13.9 ± 2.7
H. Engression	0.1240 ± 0.0002	0.4387 ± 0.0063	11.4 ± 1.7
Seq. Engression	0.1237 ± 0.0002	0.4307 ± 0.0112	29.9 ± 4.5
H. Seq. Engression	0.1235 ± 0.0003	**0.4305 ± 0.0129**	35.6 ± 6.9

Table 1: Mean prediction performance and computational cost across all model architectures on the simulation study. Time represents total duration across initialization, training, and evaluation. Extreme RMSE is evaluated on 56 observations exceeding the 99.5% training quantile. Values reported as mean ± std across 10 random seeds.

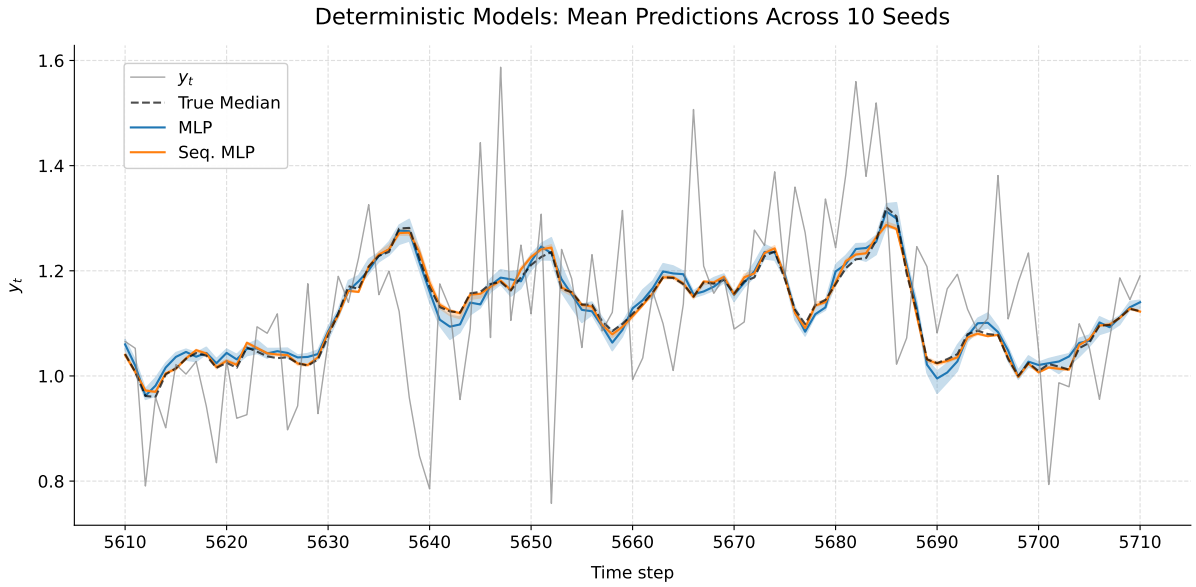

Figure 7: Deterministic models' mean predictions during a high-volatility period. For each model, solid lines show the median of mean predictions across 10 seeds, with shaded bands representing the 10th-90th percentiles of cross-seed variability.

Model	Energy Loss	PICP_{80}	Sharpness_{80}
Engression	0.0681 ± 0.0001	0.7837 ± 0.0054	0.2945 ± 0.0038
H. Engression	0.0680 ± 0.0001	0.7788 ± 0.0067	0.2937 ± 0.0050
Seq. Engression	**0.0676 ± 0.0001**	0.7883 ± 0.0067	0.3006 ± 0.0038
H. Seq. Engression	**0.0676 ± 0.0001**	**0.7898 ± 0.0115**	0.3010 ± 0.0086

Table 2: Distributional forecasting quality metrics for Engression-based models on the simulation study, with coverage (PICP) and sharpness based on 80% prediction intervals. Values reported as mean ± std across 10 random seeds.

by Sequential Engression (78.83%). By contrast, Heteroskedastic Engression exhibits a lower coverage with a 0.5 percentage point decrease over the baseline, suggesting that input-dependent noise modeling may be too unstable without clear information from the underlying process provided by a temporal encoder. In terms of sharpness, slightly wider intervals are observed in sequential models compared to the baseline. This small widening possibly reflects a more flexible distributional forecast that better captures the time-varying uncertainty of the generative process, contributing to the improved coverage while remaining below the nominal target. As with mean predictions, Heteroskedastic Sequential Engression achieves the best mean scores across metrics but exhibits substantially higher cross-seed variability in both coverage and sharpness, reflecting a consistent trade-off between performance and initialization sensitivity.

Turning to the main focus of this case study, Table 3 reports conditional quantile forecasting performance against ground truth. In terms of median prediction (Q50), all models achieve similar performance with Heteroskedastic Engression demonstrating the highest stability across seeds. By contrast, substantial improvements emerge in tail quantile predictions, where sequential architectures achieve 29-32% improvements in lower tail (Q10) and 21-27% improvements in upper tail (Q90) RMSE compared to their non-sequential counterparts. Notably, Sequential Engression achieves these gains despite using homoscedastic noise injection like the baseline, suggesting the temporal encoder captures underlying volatility dynamics through temporal aggregation as designed in the generative process. By comparison, Heteroskedastic Engression shows marginal improvements over baseline Engression (1.9% in Q10, 6.3% in Q90), indicating that input-dependent noise injection still provides slight flexibility gains, though these remain limited without the temporal context provided by sequential architectures. This improved flexibility is visible in Figure 8, where sequential models demonstrate superior ability to capture asymmetric distributional dynamics during peaks and troughs. This property is notably absent in the baseline model, which seems to construct symmetric quantile estimates across lower and upper tails, causing higher errors in asymmetric scenarios. Interestingly, while Heteroskedastic Sequential Engression achieves slightly lower performance across tail quantiles, visual inspection suggests comparable if not slightly enhanced flexibility in certain high-volatility scenarios, though at the cost of substantially higher cross-seed variability.

Collectively, these findings show that Engression-based models offer valuable alternatives to deterministic approaches. Despite requiring notably higher computational time and resources, these models maintain comparable mean performance while providing distributional forecasting capabilities and marginally improved extrapolation. Regarding

Model	RMSE Q10	RMSE Q50	RMSE Q90
Engression	0.0323 ± 0.0007	0.0181 ± 0.0008	0.0438 ± 0.0026
H. Engression	0.0317 ± 0.0020	0.0182 ± 0.0004	0.0410 ± 0.0031
Seq. Engression	**0.0220 ± 0.0006**	**0.0179 ± 0.0009**	**0.0321 ± 0.0019**
H. Seq. Engression	0.0225 ± 0.0020	0.0180 ± 0.0012	0.0323 ± 0.0035

Table 3: Quantile-specific RMSE for Engression-based models on the simulation study, evaluated against true conditional quantiles. Values reported as mean ± std across 10 random seeds.

the temporal encoder extension, effective capture of temporal dependencies in volatility dynamics was achieved, yielding substantial improvements in tail quantile predictions and distributional forecasting quality, with particularly enhanced flexibility in capturing asymmetric distributional dynamics. By contrast, the heteroskedastic extension provides marginal gains in quantile flexibility on the baseline model but comes with a substantial tradeoff in cross-seed variability and degraded coverage. When combined with sequential encoding, this extension primarily contributes to distributional forecasting quality improvements, though the performance-stability tradeoff persists, raising attention to an important consideration in model selection. Against this backdrop of increasing complexity, the original Engression remained competitive throughout the study with consistent though slightly lower performance. However, it struggled to capture asymmetric distributions where sequential and heteroskedastic dynamics are present, revealing potential limitations in complex dynamic systems.

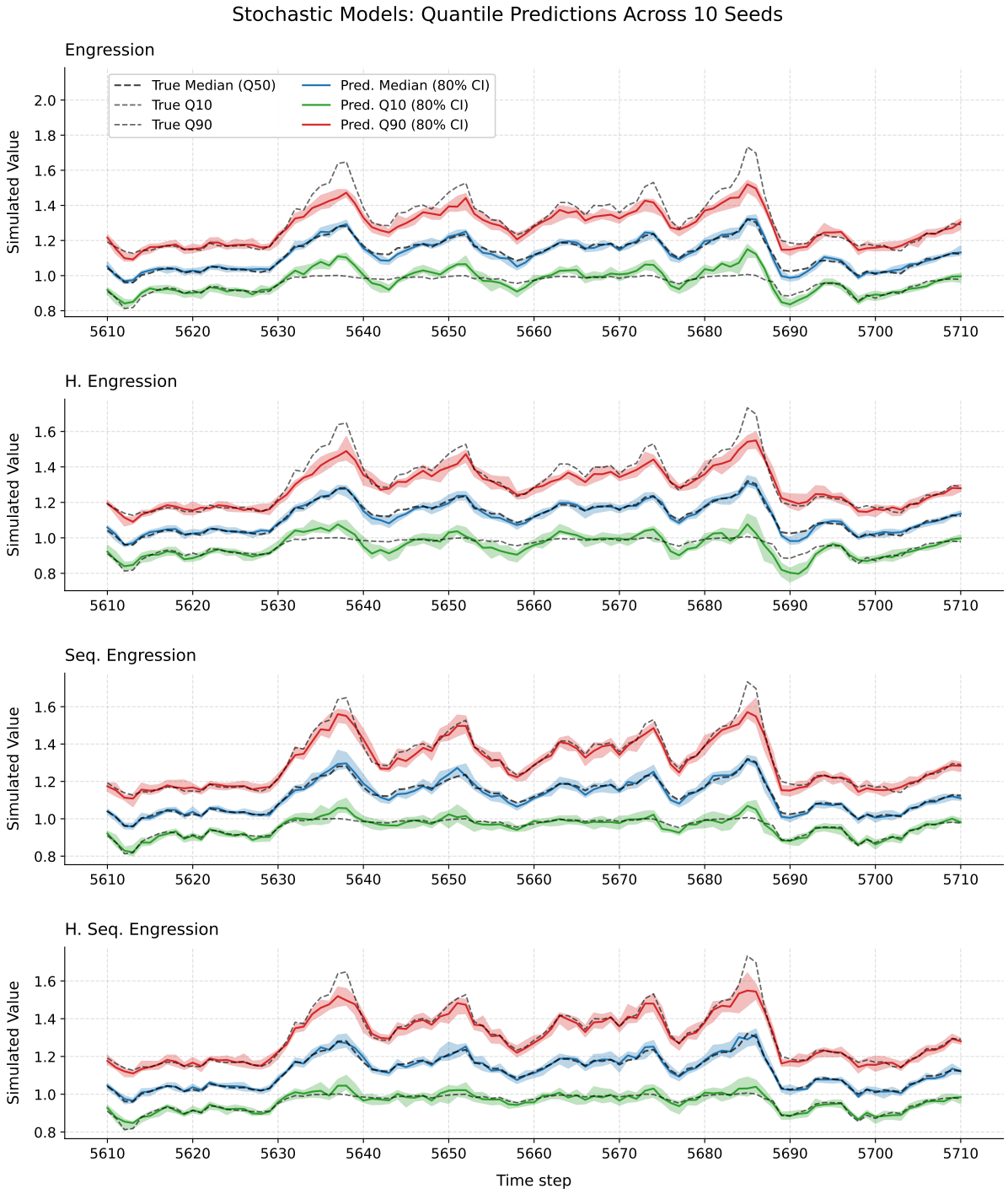

Figure 8: Engression-based models' quantile predictions during a high-volatility period. For each model, solid colored lines show the median of predicted quantiles (Q10, Q50, Q90) across 10 seeds, with shaded bands representing the 10th-90th percentiles of cross-seed variability. Black dashed lines show the corresponding true quantiles from the data generating process.

5 Application to River Discharge Forecasting

5.1 Motivation

As the second experiment of this research, we consider a real-world application on forecasting one-day-ahead river discharge levels. Through this experiment, the aim is to assess the robustness of the Engression-based models when applied to complex temporal distributions where underlying model assumptions may only partially hold. The motivation for this case study stems from the characteristics of hydrological processes, which typically exhibit sequential dependencies driven by multiple physical processes, combined with non-stationary behavior caused by climatic changes over time. As a result, these characteristics make river discharge data a challenging and informative benchmark for assessing the effectiveness of the temporal encoding and input-dependent noise mechanisms proposed in this work. Beyond these properties, river discharges offer a practical case study for flood risk assessment, where extrapolation capabilities can be assessed through the study of extreme flood events.

In order to achieve this objective, we used daily river discharge of the Aare river at the Bern-Schönau gauging station as the forecasting target, along with covariates from upstream stations across the catchment, as showcased in Figure 9. The choice of this specific dataset is motivated by its extensive use in the extreme value theory literature (Pasche and Engelke, 2024; Pasche et al., 2023; Viviroli et al., 2022), where it has been established as a reliable benchmark with well-documented data quality. This status is supported by the long-term and continuous record of daily observations spanning over 80 years, and the presence of extreme flood events including the severe August 2005 flood that caused substantial damage to the city of Bern. Collectively, these factors, along with the previously discussed complexity of hydrological processes, make this dataset a suitable case study for evaluating the proposed approaches in a challenging real-world forecasting context.

5.2 Dataset and Experimental Setup

The dataset consists of 31,046 daily observations collected continuously from January 1, 1930 to December 31, 2014 across the Aare river catchment as outlined in Figure 9, combining discharge measurements from the Swiss Federal Office for the Environment (FOEN) and precipitation data from MeteoSwiss. Each observation consists of a pair (x_t, y_t), where y_t represents the response variable as the average daily river discharge at Bern-

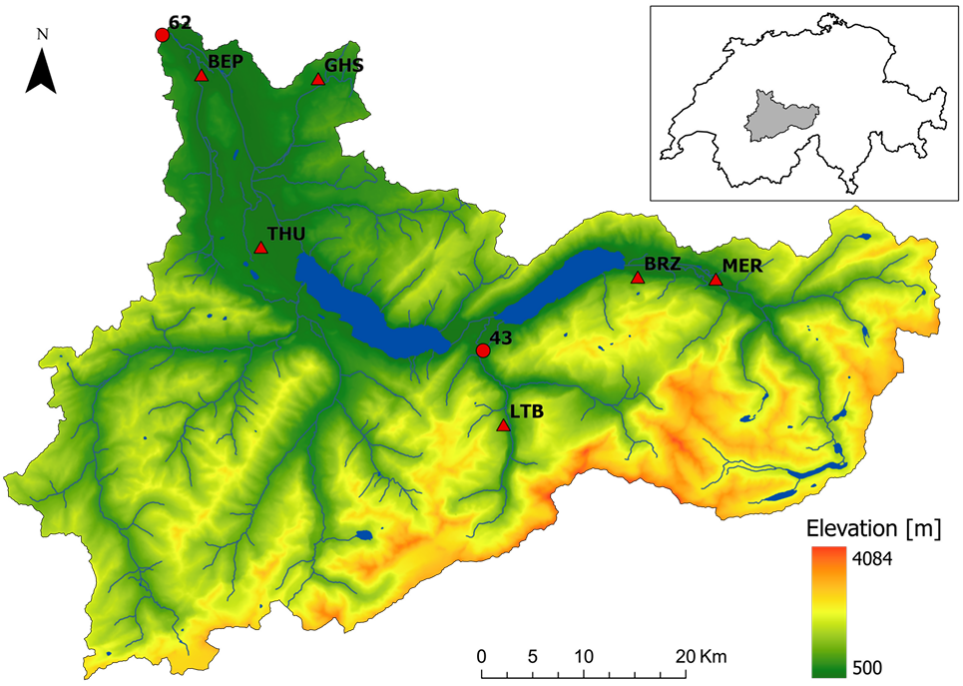

Figure 9: Topographical map of the Aare river catchment showing Bern-Schönau station 62 and upstream measurement stations. Covariates include discharge measurements from station 43 (Gsteig) and precipitation data from six stations (BEP, GHS, THU, LTB, BRZ, MER). Source: Pasche and Engelke (2024).

Schönau, and x_t includes seven upstream covariates, comprising one daily average discharge measurement from Gsteig station and six daily total precipitation measurements from meteorological stations across the catchment. The respective measurements were recorded in mm for precipitation and m^3/s for river discharge.

As illustrated in Figure 10, the response variable exhibits clear heteroskedastic behavior and seasonal patterns, reflecting the complex natural variability of Alpine river flow driven by precipitation, snowmelt, and temperature cycles across seasons. This complexity is further highlighted by the 30-day rolling standard deviation, shown in red, which showcases the time-varying volatility with visible increases during extreme flood events such as the August 2005 event shown in the lower panel. These characteristics demonstrate the non-stationary and heteroskedastic nature of the process that the proposed models are designed to capture.

Following the data preparation pipeline detailed in Section 3.2, the dataset is divided chronologically to preserve the temporal structure of the data. The resulting split is defined as follows:

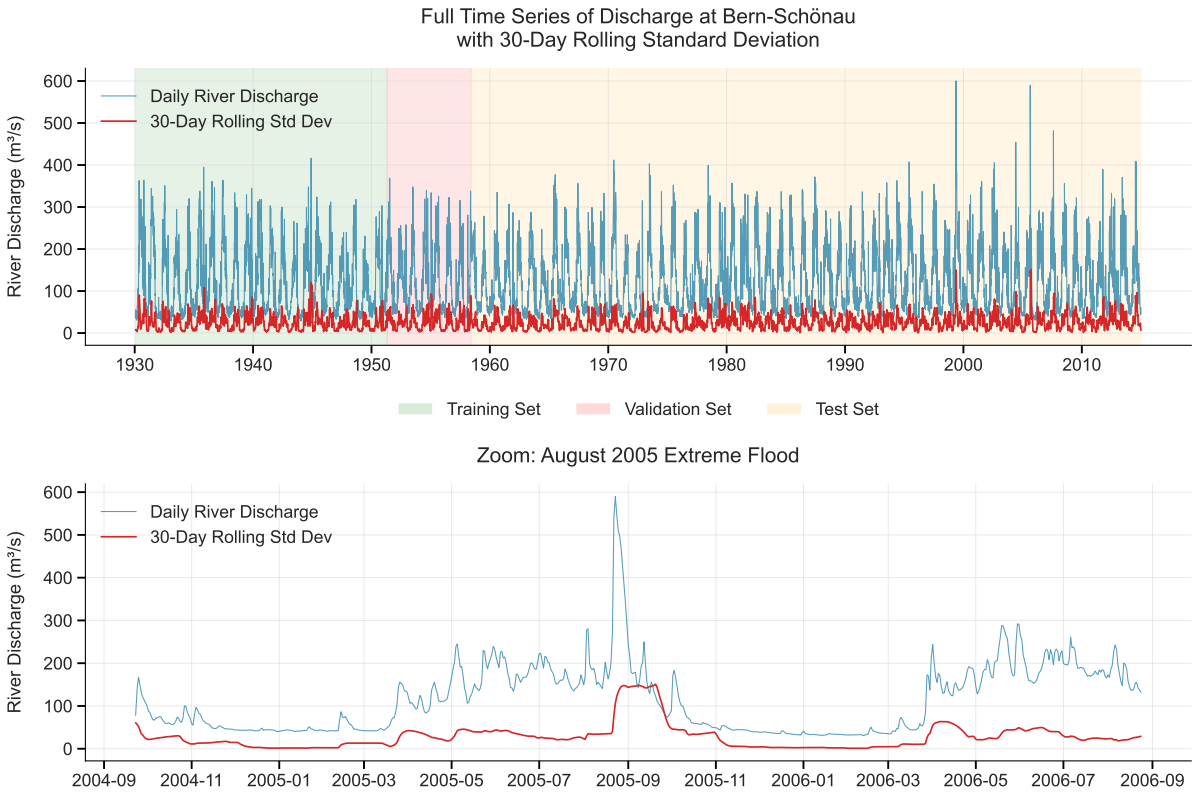

Figure 10: Daily river discharge at Bern-Schönau (1930–2014) with 30-day rolling standard deviation. The top panel shows the full time series with train/validation/test splits indicated by shaded regions. The bottom panel zooms into the August 2005 extreme flood event.

- Training set: January 1, 1930 – April 1, 1951 (7,761 observations)
- Validation set: April 2, 1951 – May 2, 1958 (2,588 observations)
- Test set: May 3, 1958 – December 31, 2014 (20,697 observations)

By this split, extreme flood events including the August 2005 event are allocated in the test set, allowing assessment of extrapolation capability on rare events unseen during training. Additionally, the extended test period enables assessment of model adaptation to long-term non-stationarity and evolving hydrological conditions. In terms of history window, $s = 10$ past observations were selected following preliminary analysis of cross-correlation patterns between lagged covariates and the response variable, along with the autocorrelation and partial autocorrelation structure of the response variable. This choice aims to provide sufficient temporal context for both feedforward and sequential architectures while reflecting the autocorrelation structure observed in the data. The resulting input space at time t is then constructed as $\tilde{x}_t = \{(x_\tau, y_\tau)\}_{\tau=t-10}^{t-1}$, following the

formalization in Section 3.1.

Reflecting the methodology and evaluation purpose of this project, no stationarity transformations or detrending procedures were applied to the data. This approach aligns with the primary objective of evaluating model robustness under non-stationary conditions and assessing the ability to extrapolate beyond the training support in the presence of distribution shifts.

Lastly, following the training and validation framework established in Section 3.5 and previously applied in Chapter 4, optimal model configurations were identified through grid search over the validation set, with hyperparameter specifications detailed in Appendix A.4.1. Subsequently, each optimal configuration was trained and evaluated across ten independent random seeds to quantify performance variability and ensure reliable comparison across model architectures.

5.3 Results

Turning to the analysis of the river discharge results, notable differences emerge in model performance compared to the controlled simulation setting. Specifically, the results provide critical insights into the robustness of the proposed models under potential misspecification and their extrapolation capability for extreme flood event forecasting.

Table 4 reports mean prediction performance across all model architectures. A notable finding emerges regarding sequential architectures, which underperform their non-sequential counterparts across both deterministic and Engression-based variants despite the inherently temporal nature of river discharge. This pattern, reflected in RMSE degradations of approximately 2-3% in mean predictions and substantially larger deterioration in extreme forecasting, suggests a limitation in how sequential models interact with the present data structure. A plausible explanation lies in the nature of the measurements, collected as daily averages. Despite the sequential nature of the underlying hydrological process, the daily aggregation may weaken the sequential signal, causing individual lag effects to dominate over temporal accumulation patterns. This interpretation is supported by the grid search results reported in Appendix A.4.1, where static attention mechanism was consistently preferred over final hidden state across sequential models, suggesting the importance of individual lag effects on top of temporal accumulation. As a result, feedforward models exploit the underlying pattern more effectively through explicit lag features under this data structure, while sequential encoding may dilute these lag-specific relationships by compressing them into recurrent hidden states.

Model	RMSE	Extreme RMSE	Time (s)
MLP	11.1349 ± 0.0829	27.9569 ± 0.9913	**33.8 ± 2.9**
Seq. MLP	11.4181 ± 0.1959	45.5817 ± 3.1949	52.8 ± 9.2
Engression	10.3152 ± 0.0745	**24.5496 ± 0.7669**	57.5 ± 5.5
H. Engression	**10.2523 ± 0.1038**	25.3863 ± 0.7183	54.2 ± 7.1
Seq. Engression	10.6258 ± 0.2982	40.4618 ± 2.8478	97.1 ± 22.4
H. Seq. Engression	10.4347 ± 0.1511	36.6335 ± 2.8430	96.4 ± 11.6

Table 4: Mean prediction performance and computational cost across all model architectures on river discharge forecasting. Time represents total duration across initialization, training, and evaluation. Extreme RMSE is evaluated on 166 observations exceeding the 99.5% training quantile. Values reported as mean ± std across 10 random seeds.

Beyond this sequential architecture challenge, Engression-based models demonstrate consistent improvements over their deterministic counterparts in both mean and extreme prediction tasks. More specifically, Engression achieves approximately 7% lower RMSE than standard MLP, with additional gains in extreme predictions of 12%. Notably, Heteroskedastic Engression provides near-equivalent performance to baseline Engression in both mean and extreme metrics, with marginal gains in RMSE (0.6%) offset by slight degradation in extreme predictions (3.4%). The most striking pattern emerges in extreme predictions, where sequential models exhibit severe deterioration with 44-65% increases in Extreme RMSE depending on architecture, highlighting the potential misspecification challenges identified earlier. Despite this overall underperformance, sequential Engression-based models still showcase notable improvements of approximately 10-20% in extreme predictions relative to Sequential MLP, demonstrating the benefit of the noise injection framework even under challenging conditions. These patterns are further outlined in Figure 11, which displays predicted versus observed extreme river discharges across all models. Regarding sequential models, a systematic underprediction appears throughout the extreme range. By contrast, all non-sequential models maintain better alignment with observations, though with slight overprediction tendencies in extreme ranges. An unexpected pattern across all models emerges in the early range, where systematic underprediction is observed despite these events being closer to the training range of observations. This underestimation may reflect the delayed reactivity inherent in next-day forecasting with daily-aggregated measurements, where rapid intra-day increases in discharge from heavy rainfall cannot be anticipated until the following forecast. Partial evidence for this interpretation appears in improved predictions in the 375-425 m^3/s range, where discharges originating from extended rainfall periods provide more gradual signal accumulation that models can anticipate and predict more effectively. In terms of cross-seed variability, Engression-based models show relatively stable predictions across seeds

Model	Energy Loss	PICP_{80}	Sharpness_{80}
Engression	4.8109 ± 0.0539	0.8118 ± 0.0087	19.3854 ± 0.5359
H. Engression	4.7701 ± 0.0462	0.7906 ± 0.0129	18.1818 ± 0.7111
Seq. Engression	4.8305 ± 0.1587	0.8173 ± 0.0135	19.2810 ± 1.1708
H. Seq. Engression	**4.6950 ± 0.0861**	**0.8070 ± 0.0154**	18.6412 ± 0.9317

Table 5: Distributional forecasting quality metrics for Engression-based models on river discharge forecasting, with coverage (PICP) and sharpness based on 80% prediction intervals. Values reported as mean ± std across 10 random seeds.

throughout most regions, while sequential variants exhibit notably higher initialization sensitivity, particularly for observations exceeding 450 m^3/s. Partial gains are further observed in Heteroskedastic Sequential Engression relative to other sequential models, reflecting marginal improvement through its input-dependent noise mechanism in capturing the volatility patterns characterizing these extreme events.

Proceeding to the distributional forecasting quality showcased in Table 5, an interesting pattern emerges regarding the heteroskedastic extension. Contrary to the mean prediction results, Heteroskedastic Sequential Engression demonstrates the best overall uncertainty quantification, achieving the lowest energy loss with 2.4% improvement over baseline Engression and near-optimal coverage at 80.7%, closest to the nominal 80% target. This finding is particularly interesting given that Sequential Engression alone underperforms baseline in both metrics, highlighting the substantial impact of the heteroskedastic extension in leveraging sequential volatility patterns being captured by the temporal encoder. Similar improvements in flexibility are observed in Heteroskedastic Engression, which achieves modest gains in energy loss (0.8% better than baseline) alongside the sharpest prediction intervals, though at the cost of slight under-coverage. Compared to previous metrics, cross-seed variability shows different patterns, with Heteroskedastic Sequential Engression exhibiting highest initialization sensitivity in coverage but reasonable stability in energy loss despite its complexity.

To assess the practical viability of the proposed models for early warning applications, the August 2005 extreme flood event is examined as a critical test case for evaluating alarm system capabilities. Analyzing the mean predictions across models presented in Figure 12, the systematic delayed reactivity discussed earlier becomes evident, with all models exhibiting approximately one day lag during the rapid discharge increase. This delay, inherent to the next-day forecasting framework with daily-aggregated measurements, raises concerns regarding their viability for practical use in flood early warning systems. Among non-sequential models, all variants exhibit similar prediction patterns with comparable performance during both rise and decline phases, though slight overestimation is observed

during the peak period. Regarding cross-seed stability, MLP demonstrates notably higher initialization sensitivity compared to Engression-based models, which maintain more stable predictions across seeds. In contrast, sequential variants demonstrate the systematic underprediction identified in Figure 11, with marginal gains visible in Engression-based models that better track the gradual descent period. Turning to distributional forecasts in Figure 13, substantial differences emerge in uncertainty quantification capabilities. Both heteroskedastic variants stand out by exhibiting wide and adaptive prediction intervals around the peak, appropriately reflecting elevated uncertainty during high-volatility periods and enabling more informative risk assessment. Homoskedastic models, conversely, maintain narrow intervals even during the extreme event, suggesting insufficient uncertainty calibration during extreme periods and limited capability for the task.

Taken together, these results demonstrate the value that the pre-additive noise approach provides despite challenges posed by real-world data complexity. Across all model families, Engression-based variants consistently outperform their deterministic counterparts in both mean and extreme predictions, with this robustness maintained even when sequential architectures underperform due to structural misspecification. Despite this limitation in extrapolation, sequential extensions reveal an intriguing capability to capture volatility signals that, along with the heteroskedastic extension, offer the best distributional capability among the models analyzed. By comparison, the simpler Heteroskedastic Engression achieves a more favorable balance, providing stability in extrapolation while achieving competitive adaptive uncertainty quantification via the heteroskedastic extension, as showcased during the August 2005 extreme event. As a result, while the proposed extensions demonstrate clear improvements over the baseline model, they reveal fundamental trade-offs in model selection between distributional quality and extrapolation capability inherent to their respective architectural designs.

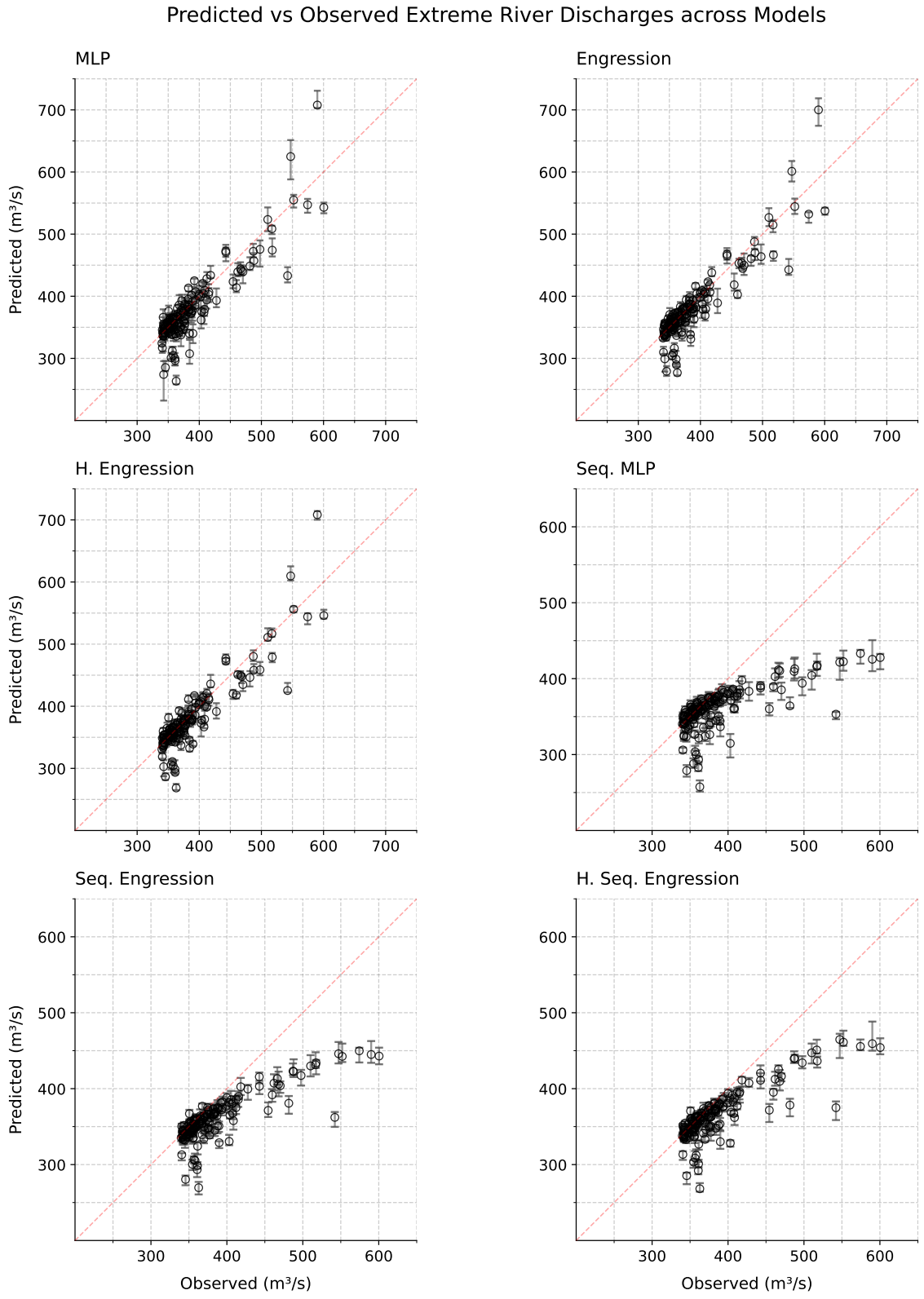

Figure 11: Mean predictions versus observed values for extreme river discharges (>99.5% training quantile) across all model architectures (166 observations). Each point represents the median prediction across 10 random seeds, with vertical error bars showing the 80% interval (10th-90th percentiles) of cross-seed variability.

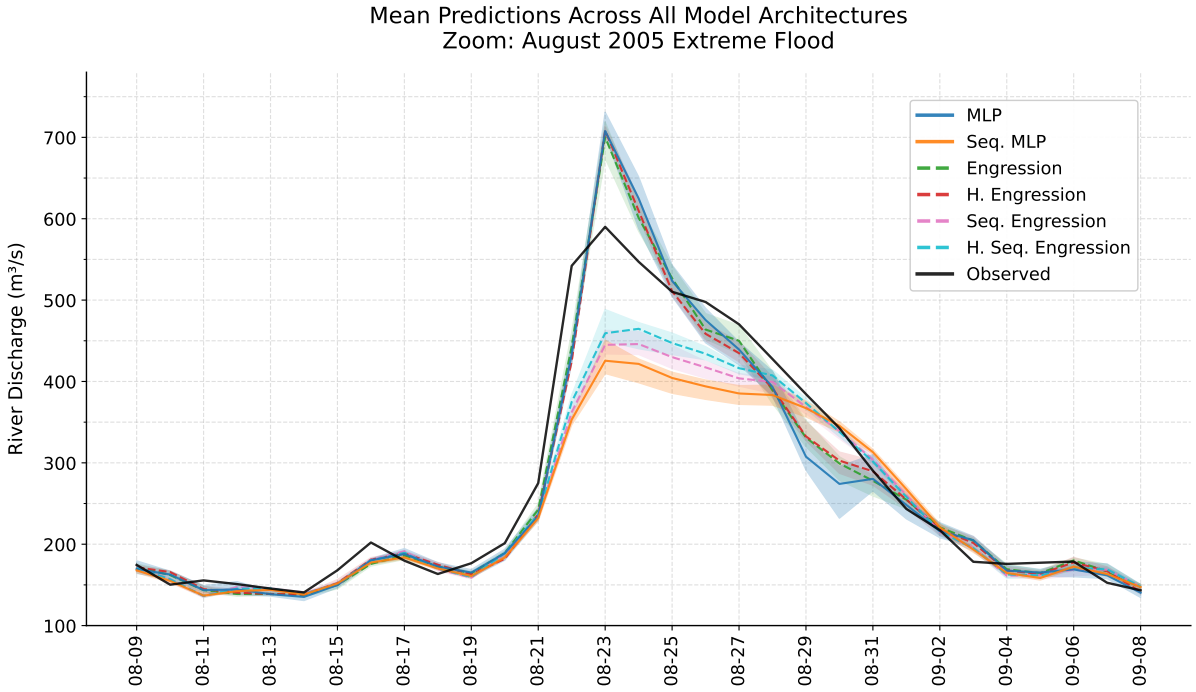

Figure 12: Mean predictions across all model architectures during the August 2005 extreme flood event at Bern-Schönau. Lines show median predictions across 10 random seeds, with shaded regions indicating cross-seed variability (10th-90th percentiles).

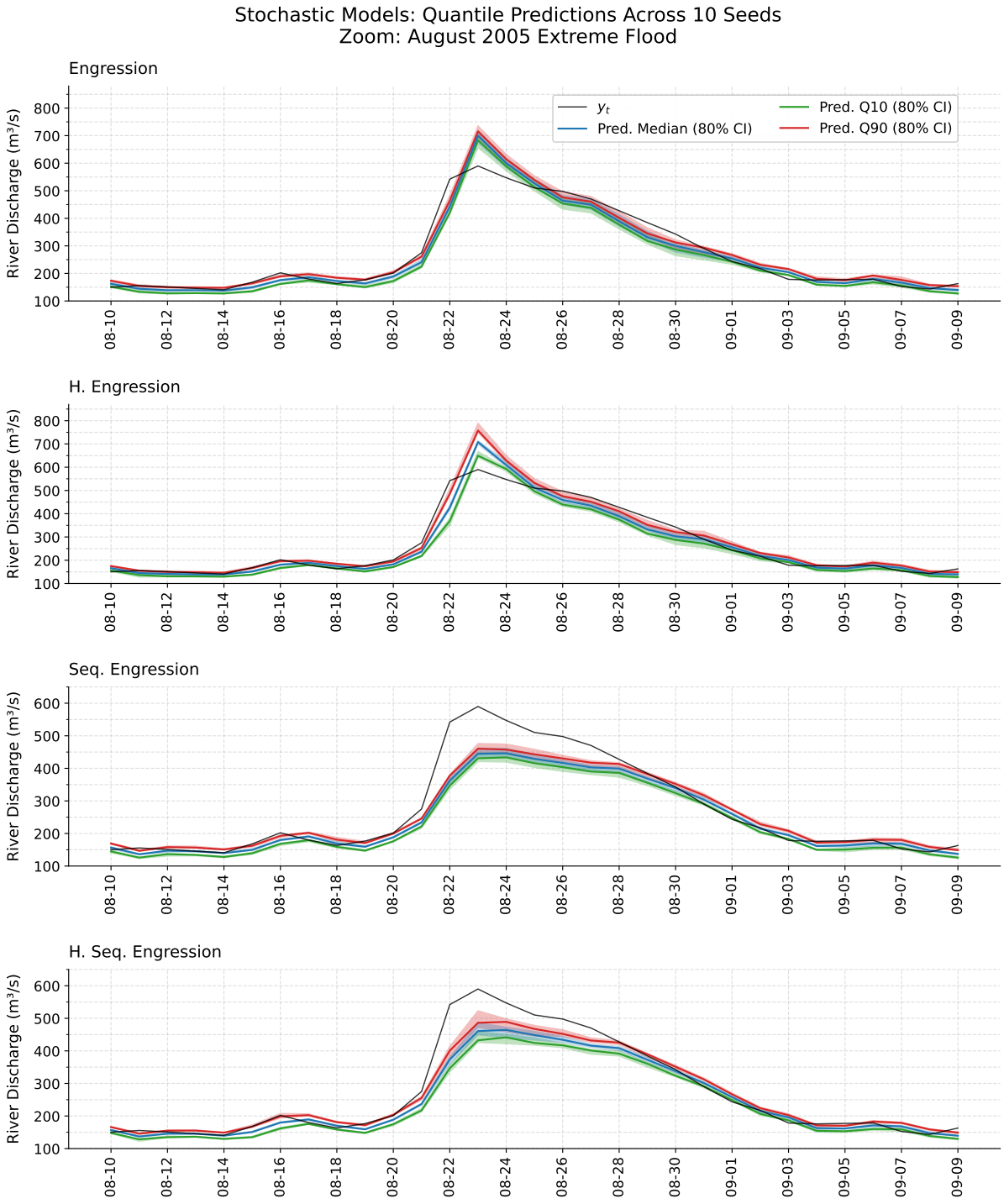

Figure 13: Engression-based models' quantile predictions during the August 2005 extreme flood event at Bern-Schönau. For each model, solid colored lines show the median of predicted quantiles (q10, q50, q90) across 10 seeds, with shaded bands representing the 10th-90th percentiles of cross-seed variability.

6 Discussion

6.1 Summary of Findings

Building on the simulation study presented in Chapter 4 and the real application analysis in Chapter 5, this section discusses the common patterns observed across the evaluated models.

Among the results, the most important finding concerns the significant advantages demonstrated by Engression-based models over their deterministic counterparts. Across both settings, Engression-based models achieved competitive and often superior mean predictions while demonstrating significant gains in extrapolation capabilities. In the river discharge study, Engression variants exhibited approximately 7-9% lower RMSE than their deterministic counterparts, with further improvements of 9-20% in extreme predictions. The simulation results showed more modest gains of 1-2% in extrapolation capability, largely due to the noisier setting in those extreme observations, while maintaining comparable mean prediction performance. Importantly, as evidenced in the river discharge study, these improvements persisted even when the model structure was suboptimally aligned with the underlying data, underscoring the robustness and flexibility of the pre-additive noise mechanism. Together, these results seems to validate the improved accuracy of Engression-based models in contexts where accurate mean trend estimation is equally important as reliable uncertainty quantification and enhanced extreme event prediction.

Regarding the individual extensions, both the temporal encoder and conditional noise encoder exhibited strong context-dependent performance determined by the nature of the underlying mean and volatility dynamics. Sequential models achieved substantial tail quantile improvements in the simulation setting, successfully capturing asymmetric distributions where the baseline model produced symmetric forecasts. However, they exhibited marked underperformance in mean and extreme predictions in the river discharge study, where daily aggregation weakened sequential signals in favor of stronger lag-specific contributions. The heteroskedastic extension revealed more nuanced dependencies tied to volatility's driving forces. In the simulation setting, heteroskedastic patterns seemed to appear largely embedded within sequential dynamics, making the sequential encoder alone sufficient and rendering the additional complexity of heteroskedastic extension mostly redundant. Conversely, the river discharge study exhibited complex heteroskedastic signals with both sequential and non-sequential components, allowing each extension to contribute distinct information. This duality manifested as marginal improvements in distributional quality achieved by the heteroskedastic extension over the baseline, with

further gains observed when combined with sequential encoding, indicating additional volatility dynamics captured through the temporal structure.

From a practical perspective, the marginal gains introduced by these extensions came with notable costs in their implementation. Across both experiments, the increased flexibility from architectural complexity often led to heightened cross-seed variability in the forecasts, partially compromising the performance stability that characterized the original model. The computational burden proved equally significant, with Engression-based extensions requiring up to 3 times more processing time than their deterministic counterparts, raising legitimate concerns about implementation feasibility in resource-constrained environments.

Additional findings emerged from the proposed architectural design choices evaluated in the grid search. Among them, the vectorized noise representation was consistently preferred over the scalar alternative, demonstrating the benefit of modeling different noise dimensions compared to a single scalar as used in the original model under fixed value. Similarly, the static attention mechanism was always favored over the final hidden state approach in both deterministic and Engression-based architectures, confirming the limitations of standard GRU architectures in modeling complex temporal dynamics. These consistent preferences validate the strategic design decisions made to improve the original architecture's adaptability in time series forecasting.

In light of these findings, model selection requires careful consideration of the underlying data characteristics. While baseline Engression demonstrated consistent robustness across both settings, the proposed extensions showed context-dependent performance that hinges on the presence of specific patterns in the data. This underscores the importance of aligning architectural choices with observable data characteristics, particularly given the complexity of underlying patterns commonly observed in temporal forecasting applications.

6.2 Limitations

The analysis in this thesis contains several limitations that should be acknowledged when interpreting the results.

The first limitation concerns the extension of the empirical validation, which was restricted to two case studies comprising a controlled simulation and a river discharge forecasting task. While these settings provided complementary insights under different conditions, broader validation across diverse time series domains would strengthen generalizability claims. This is particularly relevant for applications with stronger sequential patterns, where the sequential models' capabilities could be more thoroughly assessed.

The second limitation concerns computational resource constraints that affected several methodological choices throughout this work. Most notably, batch sizes were limited to values smaller than ideal, as discussed in Section 3.5, potentially affecting training stability and convergence. Additionally, the hyperparameter grid search space was necessarily constrained due to the exponential growth in possible configurations, meaning better architectural setups may have existed but were not discovered within the explored range.

Lastly, the simulation design incorporated multiple interacting features of sequential dependencies, heteroskedasticity, and the pre-additive noise structure to allow a comprehensive evaluation of the proposed extensions under theoretical assumption. However, this complexity likely introduced interactions between features that complicated the isolation of individual architectural contributions, even for models optimized for those specific patterns. A more controlled approach using simpler experiments that isolate individual features could provide clearer insights into each extension's specific contribution, though this would come at the cost of reduced realism in the simulated dynamics.

6.3 Future Work

This work opens several directions for future research to deepen understanding of distributional approaches for temporal forecasting.

First and foremost, with greater computational resources, the limitations identified earlier could be systematically addressed to better assess the models' full potential. Specifically, larger sample sizes for conditional distribution estimation would enable more reliable quantile forecasting across the full distribution range, particularly in the tails. This would provide clearer insights into the stability of distributional forecasts and further highlight the differences in asymmetric distribution modeling capabilities between the proposed extensions and baseline Engression. Additionally, systematic comparison with other distributional forecasting approaches in time series, such as quantile regression methods or other probabilistic models, would provide valuable context for the relative strengths of Engression-based approaches. Similarly, additional benchmarking against established theoretical time series models would further validate the practical utility of the proposed framework.

Alternatively, different temporal encoding architectures deserve investigation. Among them, one promising direction is the application of Feedforward Sequential Memory Networks (Zhang et al., 2017), which offer a feedforward structure with learnable memory blocks. These networks have demonstrated competitive performance in long sequence modeling while avoiding backpropagation through time, thereby reducing computational

training costs. This extension would enable evaluation of the proposed framework on high-frequency temporal data with stronger long-range dependencies, such as financial tick data or sensor measurements collected at fine temporal intervals.

More broadly, the pre-additive noise framework could be explored beyond time series applications to establish its viability as a general approach for conditional distributional modeling. For instance, applying this framework to image classification tasks using convolutional neural networks as encoders (LeCun et al., 1998) would test whether the extrapolation and uncertainty quantification benefits observed in temporal settings generalize to other data structures.

7 Conclusion

In conclusion, this study demonstrates the successful extension of the Engression framework to temporal forecasting, showcasing the potential of pre-additive noise modeling in sequential settings. By integrating temporal encoders and heteroskedastic noise mechanisms, the proposed framework achieved consistent improvements in uncertainty quantification while maintaining the extrapolation benefits of the pre-additive design. This is particularly evident in the performance demonstrated across both experimental settings, where Engression-based models achieved competitive mean prediction performance alongside superior extrapolation capabilities compared to their deterministic counterparts. While the individual proposed extensions provided context-dependent benefits influenced by underlying data characteristics, each component effectively captured distinct signals in the data, with amplified effects when combined. Specifically, the temporal encoder proved valuable for modeling sequential volatility patterns, while the heteroskedastic extension enabled adaptive uncertainty modeling during high-volatility periods. Together, these extensions addressed key limitations of the baseline model in handling asymmetric distributions and time-varying uncertainty. Overall, the findings establish this extended framework as a promising direction for distributional temporal forecasting, demonstrating that pre-additive noise modeling can deliver reliable uncertainty quantification without the constraints of restrictive parametric assumptions.

Acknowledgments

I would like to sincerely thank Prof. Dr. Sebastian Engelke and PhD candidate Olivier Pasche for proposing this research topic and guiding me throughout this research journey. Their invaluable support and feedback was always available during the entire process, making this work a truly enriching learning experience.

Warm thanks are also extended to my colleagues and family members who have been there through the ups and downs of this project. Their solidarity, assistance, and constant support made this work possible.

Declaration

The author declares that AI tools, including Claude, GitHub Copilot, and Overleaf's AI integrated features, were used during the preparation of this thesis for language editing, proofreading, and code optimization. These tools were used solely to enhance clarity and efficiency. All research design, analysis, interpretation, and original content remain entirely the work of the author, developed in consultation with the thesis supervisor.

References

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

L. Bottou. Large-scale machine learning with stochastic gradient descent. In Y. Lechevallier and G. Saporta, editors, *Proceedings of COMPSTAT'2010*, pages 177–186. Physica-Verlag HD, 2010.

G. E. P. Box and G. C. Tiao. Intervention analysis with applications to economic and environmental problems. *Journal of the American Statistical Association*, 70(349):70–79, 1975.

G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung. *Time Series Analysis: Forecasting and Control*. John Wiley & Sons, 5th edition, 2015.

K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning Phrase Representations Using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734. Association for Computational Linguistics, 2014.

J. Dancker. A brief introduction to recurrent neural networks, 2022. [Online]. Available: https://towardsdatascience.com/a-brief-introduction-to-recurrent-neural-networks-638f64a61ff4. Accessed: 2025-08-04.

C. Dugas, Y. Bengio, F. Bélisle, C. Nadeau, and R. Garcia. Incorporating second-order functional knowledge for better option pricing. In *Advances in Neural Information Processing Systems 13 (NIPS 2000)*, pages 472–478. MIT Press, 2001.

J. L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

F. Furizal, A. Fawait, H. Maghfiroh, A. Ma'arif, A. Firdaus, and I. Suwarno. Long Short-Term Memory vs Gated Recurrent Unit (GRU): A literature review on the performance of deep learning methods in temperature time series forecasting. *International Journal of Robotics and Control Systems*, 4(3):1506–1526, 2024.

F. Galton. Regression Towards Mediocrity in Hereditary Stature. *The Journal of the Anthropological Institute of Great Britain and Ireland*, 15:246–263, 1886.

X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *Proceedings of AISTATS 2011*, pages 315–323. PMLR, 2011.

T. Gneiting and M. Katzfuss. Probabilistic forecasting. *Annual Review of Statistics and Its Application*, 1:125–151, 2014.

T. Gneiting and A. E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

D. P. Helmbold and P. M. Long. Surprising properties of dropout in deep networks. In *Proceedings of the 2017 Conference on Learning Theory*, volume 65 of *Proceedings of Machine Learning Research*, pages 1123–1146. PMLR, 2017.

S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8): 1735–1780, 1997.

D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization, 2014. URL `https://arxiv.org/abs/1412.6980`. arXiv preprint arXiv:1412.6980 [cs.LG].

R. Koenker and G. Bassett. Regression Quantiles. *Econometrica*, 46(1):33–50, 1978.

A. Krogh and J. Hertz. A simple weight decay can improve generalization. In J. Moody, S. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems*, volume 4, pages 950–957. Morgan Kaufmann, 1991.

X.-H. Le, H. V. Ho, G. Lee, and S. Jung. Application of Long Short-Term Memory (LSTM) neural network for flood forecasting. *Water*, 11(7):1387, 2019.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

M. Mirza and S. Osindero. Conditional Generative Adversarial Nets, 2014. URL `https://arxiv.org/abs/1411.1784`. arXiv preprint arXiv:1411.1784 [cs.LG].

O. C. Pasche and S. Engelke. Neural networks for extreme quantile regression with an application to forecasting of flood risk. *The Annals of Applied Statistics*, 18(4): 1727–1747, 2024.

O. C. Pasche, V. Chavez-Demoulin, and A. C. Davison. Causal modelling of heavy-tailed variables and confounders with application to river flow. *Extremes*, 26(3):573–594, 2023.

K. Pearson and O. M. F. E. Henrici. Vii. mathematical contributions to the theory of evolution. iii. regression, heredity, and panmixia. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 187:253–318, 1896.

L. Prechelt. Early stopping – but when? In G. B. Orr and K.-R. Müller, editors, *Neural Networks: Tricks of the Trade*, pages 55–69. Springer Berlin Heidelberg, 1998.

X. Shen and N. Meinshausen. Engression: Extrapolation through the lens of distributional regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 86(2):305–329, 2024.

K. Sohn, H. Lee, and X. Yan. Learning structured output representation using deep conditional generative models. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28 (NeurIPS 2015)*, pages 3483–3491. Curran Associates Inc., 2015.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

L. J. Tashman. Out-of-sample tests of forecasting accuracy: An analysis and review. *International Journal of Forecasting*, 16(4):437–450, 2000.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, volume 30, pages 6000–6010. Curran Associates, Inc., 2017.

D. Viviroli, A. E. Sikorska-Senoner, G. Evin, M. Staudinger, M. Kauzlaric, J. Chardon, A.-C. Favre, B. Hingray, G. Nicolet, D. Raynaud, J. Seibert, R. Weingartner, and C. Whealton. Comprehensive space-time hydrometeorological simulations for estimating very rare floods at multiple sites in a large river basin. *Natural Hazards and Earth System Sciences*, 22(9):2891–2920, 2022.

A. Yunita, M. I. Pratama, M. Z. Almuzakki, H. Ramadhan, E. A. P. Akhir, A. B. F. Mansur, and A. H. Basori. Performance analysis of neural network architectures for time series forecasting: A comparative study of RNN, LSTM, GRU, and hybrid models. *MethodsX*, 15:103462, 2025.

S. Zhang, C. Liu, H. Jiang, S. Wei, L. Dai, and Y. Hu. Nonrecurrent neural structure for long-term dependence. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(4):871–884, 2017.

A Appendix

A.1 Additional Architecture Diagrams

This appendix includes additional architectural diagrams (Figures 14–16) for the baseline and intermediate model variants described in Section 3.4.

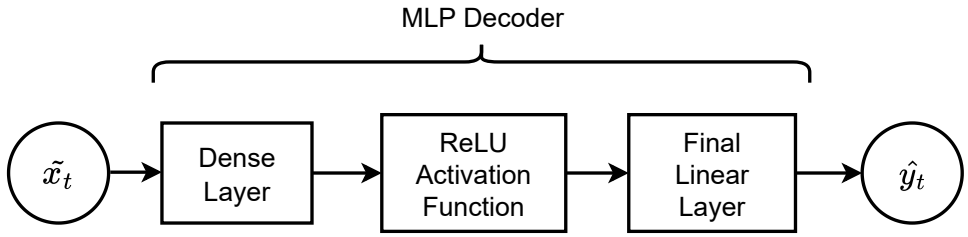

Figure 14: Diagram of the MLP architecture, serving as the deterministic counterpart of Engression.

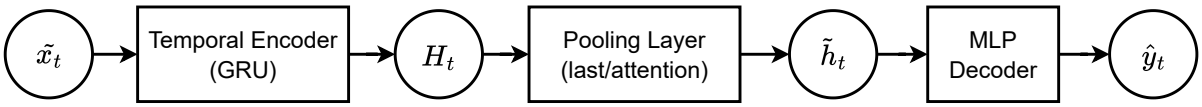

Figure 15: Diagram of the Sequential MLP architecture, serving as the deterministic counterpart of Sequential Engression with a pooled temporal encoder and a standard MLP decoder.

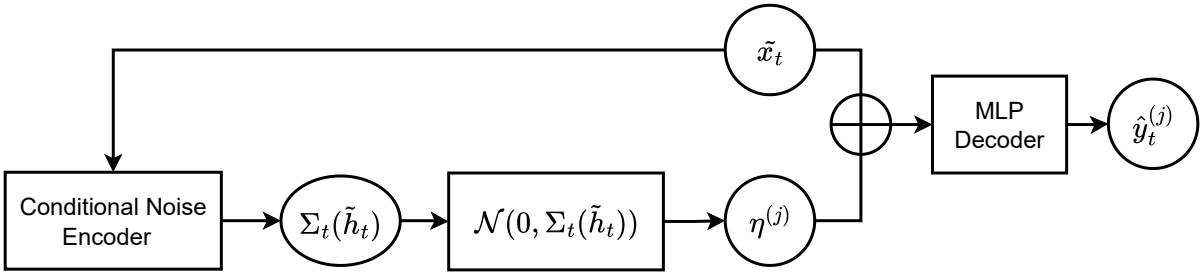

Figure 16: Diagram of the Heteroskedastic Engression architecture, extending Engression with input-dependent noise injection.

A.2 Search Spaces and Configurations

Based on the hyperparameter grid and its applicability across models classes as detailed in Table 6, the grid search results in the following number of configurations per model:

- 20 configurations for MLP and Engression
- 72 configurations for Sequential MLP and Sequential Engression
- 40 configurations for Heteroskedastic Engression
- 144 configurations for Heteroskedastic Sequential Engression

Hyperparameter	**Candidate Values**	**Applicable Models**
MLP Decoder Architecture	- 64 - 128 - 64-64 - 128-64 - 64-64-64	- MLP - Engression - Heteroskedastic Engression
GRU Encoder Architecture	- 64 - 128 - 64-64	- Sequential MLP - Sequential Engression - Heteroskedastic Sequential Engression
Sequential MLP Decoder Architecture	- 64 - 32-32 - 64-64	- Sequential MLP - Sequential Engression - Heteroskedastic Sequential Engression
Sequential Pooling Mechanism	- Last hidden state - Static attention mechanism	- Sequential MLP - Sequential Engression - Heteroskedastic Sequential Engression
Vector Noise Dimension (d_η)	- 100	- All Engression-based models
Homoskedastic Noise (σ^2)	- 1.0	- Engression - Sequential Engression
Heteroskedastic Noise Representation	- Scalar - Vectorized	- Heteroskedastic Engression - Heteroskedastic Sequential Engression
Learning Rate	- 1×10^{-3} - 5×10^{-4}	- All models
Weight Decay	- 0 - 1×10^{-4}	- All models

Table 6: Hyperparameter grids and applicability across models in both experiments. Architecture candidate values represent the hidden dimensions of each layer.

A.3 Supplementary Results: Simulation Study

A.3.1 Grid Search Results

The grid search was conducted over the hyperparameter ranges reported in Appendix A.2, using a fixed random seed across all hyperparameter combinations and model classes. Early stopping on the validation set was applied every other epoch with an improvement threshold of $\Delta = 5 \times 10^{-4}$ for all models. For the Engression-based models, patience was set to 15 validation checks to account for the higher variability induced by the stochastic energy loss. For deterministic models, by contrast, a shorter patience of 10 checks was used to reflect the smoother and less noisy MSE validation curve. All validation losses reported below are on the standardized scale used during training. Additionally, a batch size of 256 was used across all families of models. The resulting optimal architectures and training parameters are summarized below:

- **MLP**: A single-layer architecture with 64 hidden units, trained with learning rate 5×10^{-4} and weight decay 1×10^{-4}. The best MSE validation loss achieved was 0.8219.
- **Sequential MLP**: A single-layer GRU encoder with hidden size 64, followed by a two-layer MLP with sizes [32, 32] and pooling via a static attention mechanism. Training used learning rate 5×10^{-4} and weight decay 1×10^{-4}. The best MSE validation loss achieved was 0.8177.
- **Engression**: A single-layer MLP with 64 units, trained with learning rate 5×10^{-4} and weight decay 1×10^{-4}. The best Energy validation loss achieved was 0.4949.
- **Heteroskedastic Engression**: A single-layer MLP with 64 units and vectorized noise representation, trained with learning rate 5×10^{-4} and without weight decay. The best Energy validation loss achieved was 0.4940.
- **Sequential Engression**: A single-layer GRU encoder with hidden size 64, followed by a single-layer MLP with 64 units and pooling via a static attention mechanism. Training used learning rate 1×10^{-3} and without weight decay. The best Energy validation loss achieved was 0.4900.

- **Heteroskedastic Sequential Engression**: A two-layer GRU encoder with hidden sizes [64, 64], followed by a single-layer MLP with 64 units, pooling via a static attention mechanism and vectorized noise representation. Training used learning rate 5×10^{-4} and without weight decay. The best Energy validation loss achieved was 0.4893.

Additional visualization of the validation loss distributions across all tested hyperparameter configurations for each model class is provided in Figure 17, showing the range of performance within the explored grid search space.

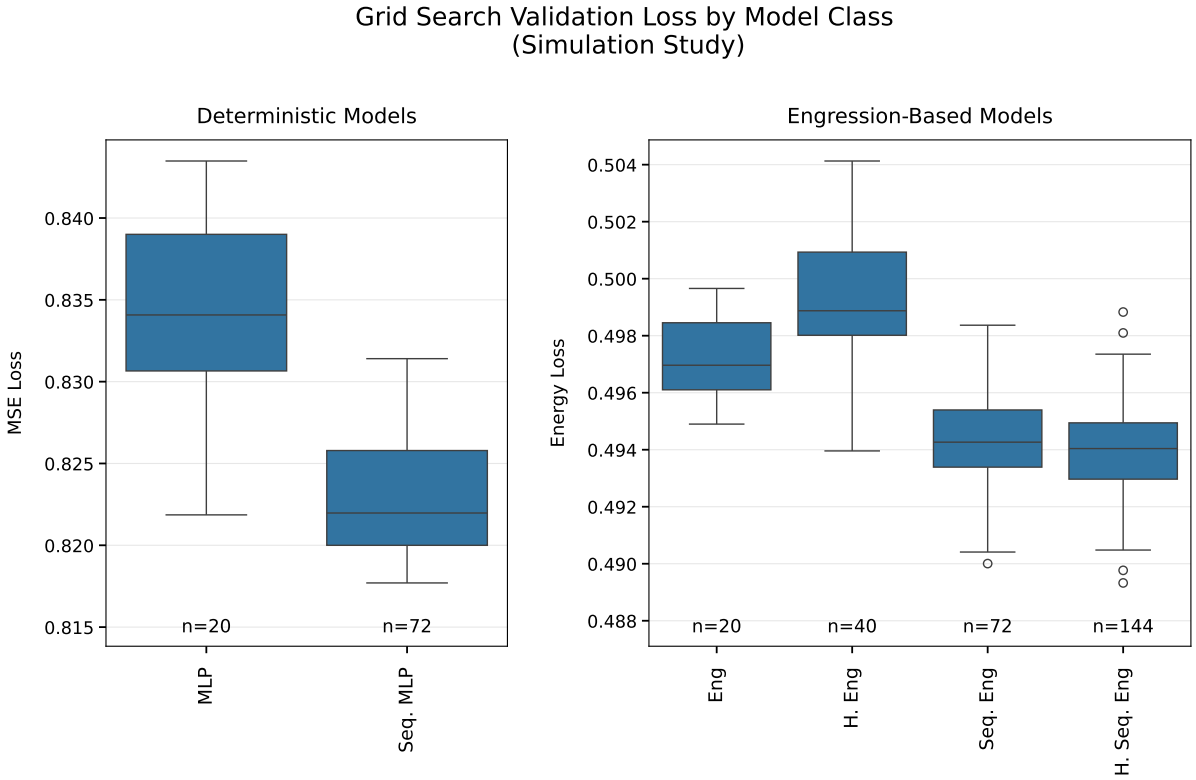

Figure 17: Box plots showing validation loss distributions (on standardized scale) across all hyperparameter combinations tested for each model in the simulation study.

A.3.2 Additional Visualizations

This section provides supplementary visualizations of distributional predictions and extreme event performance in the simulation study. Figure 18 shows mean predictions along with quantile forecasts during a high-volatility period, while Figure 19 examines mean predictions for extreme observations across the different models evaluated.

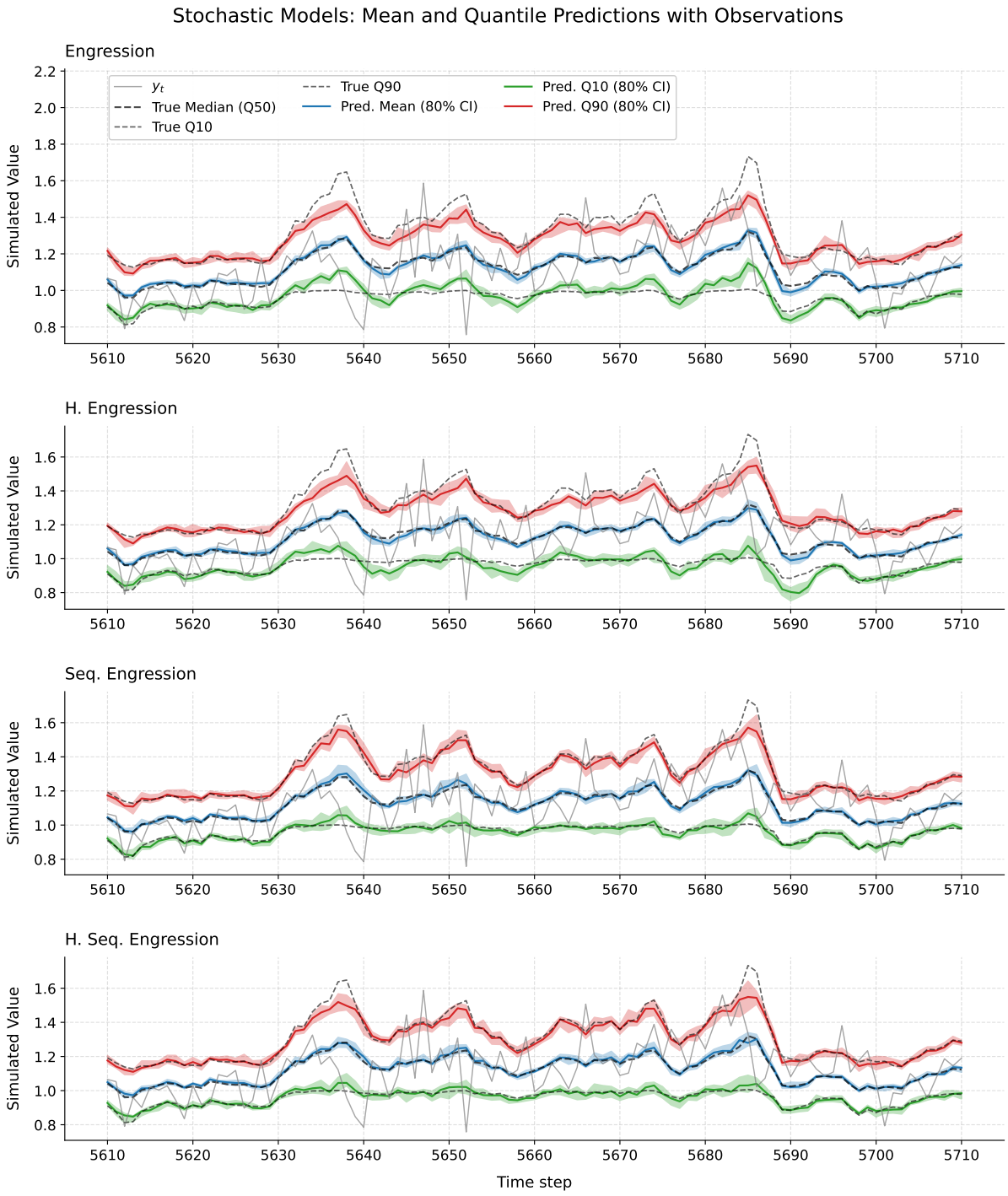

Figure 18: Engression-based models' mean and quantile predictions during a high-volatility period. For each model, solid colored lines show the median of predicted mean and quantiles (q10, q90) across 10 seeds, with shaded regions indicating cross-seed variability (10th-90th percentiles). Black dashed lines show the corresponding true quantiles from the data generating process.

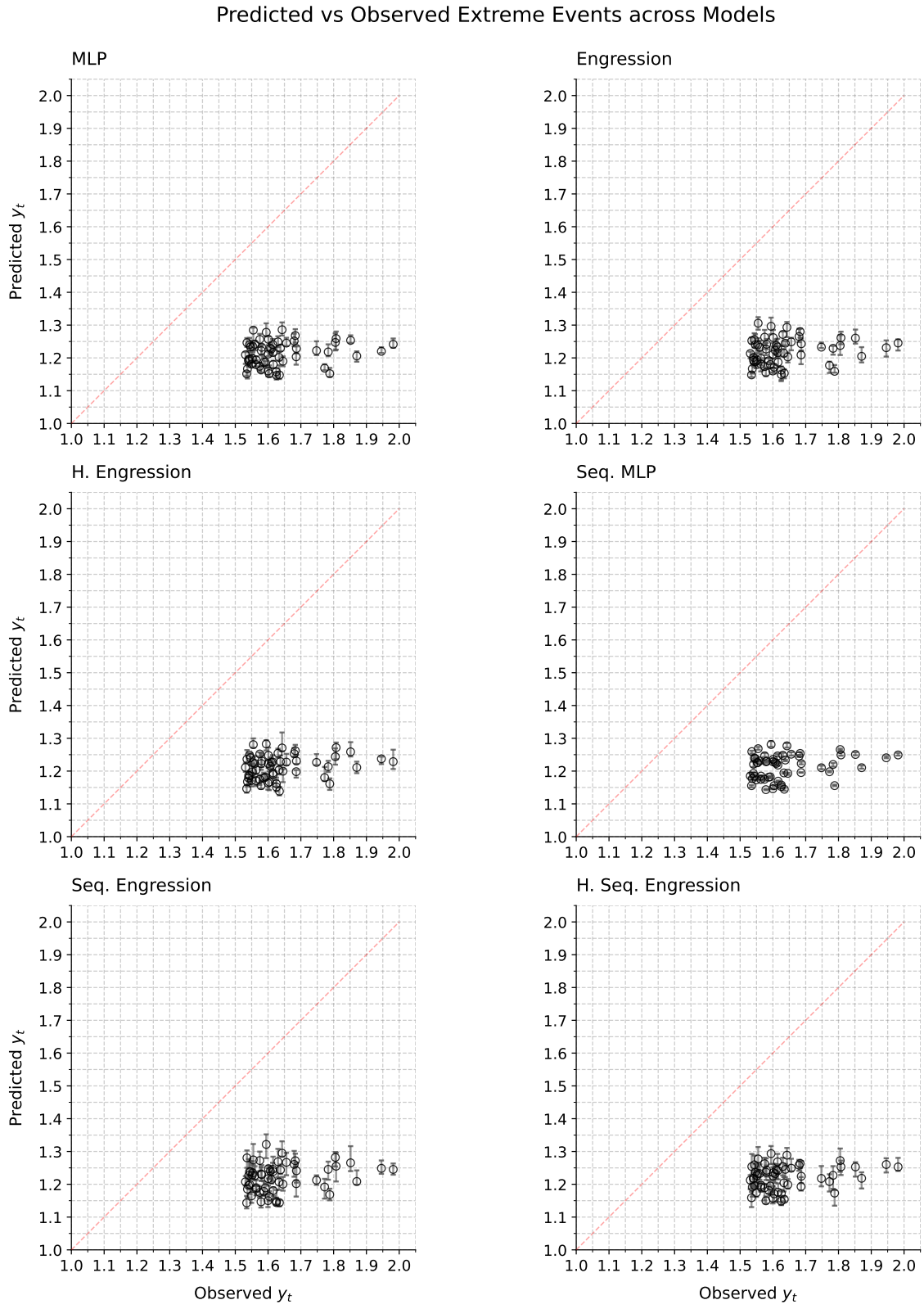

Figure 19: Mean predictions versus observed values for extreme event (>99.5% training quantile) across all model architectures (56 observations). Each point represents the median prediction across 10 random seeds, with vertical error bars showing the 80% interval (10th-90th percentiles) of cross-seed variability.

A.4 Supplementary Results: River Discharge Study

A.4.1 Grid Search Results

Following the same grid search methodology, with early stopping thresholds and patience settings as specified in Appendix A.3.1, the optimal model configurations on the standardized validation scale are summarized below:

- **MLP**: A single-layer architecture with 64 hidden units, trained with learning rate 5×10^{-4} and without weight decay. The best MSE validation loss achieved was 0.0157.
- **Sequential MLP**: A single-layer GRU encoder with hidden size 128, followed by a two-layer MLP with sizes [64, 64] and pooling via a static attention mechanism. Training used learning rate 1×10^{-3} and weight decay 1×10^{-4}. The best MSE validation loss achieved was 0.0127.
- **Engression**: A two-layer MLP with hidden sizes [64, 64], trained with learning rate 5×10^{-4} and weight decay 1×10^{-4}. The best Energy validation loss achieved was 0.0533.
- **Heteroskedastic Engression**: A single-layer MLP with 64 units and vectorized noise representation, trained with learning rate 5×10^{-4} and weight decay 1×10^{-4}. The best Energy validation loss achieved was 0.0533.
- **Sequential Engression**: A single-layer GRU encoder with hidden size 128, followed by a single-layer MLP with 64 units and pooling via a static attention mechanism. Training used learning rate 1×10^{-3} and without weight decay. The best Energy validation loss achieved was 0.0530.
- **Heteroskedastic Sequential Engression**: A single-layer GRU encoder with hidden size 128, followed by a single-layer MLP with 64 units, pooling via a static attention mechanism and vectorized noise representation. Training used learning rate 1×10^{-3} and without weight decay. The best Energy validation loss achieved was 0.0511.

The validation loss distributions across all tested hyperparameter configurations for each model class are visualized in Figure 20, showing the performance range within the explored grid search space.

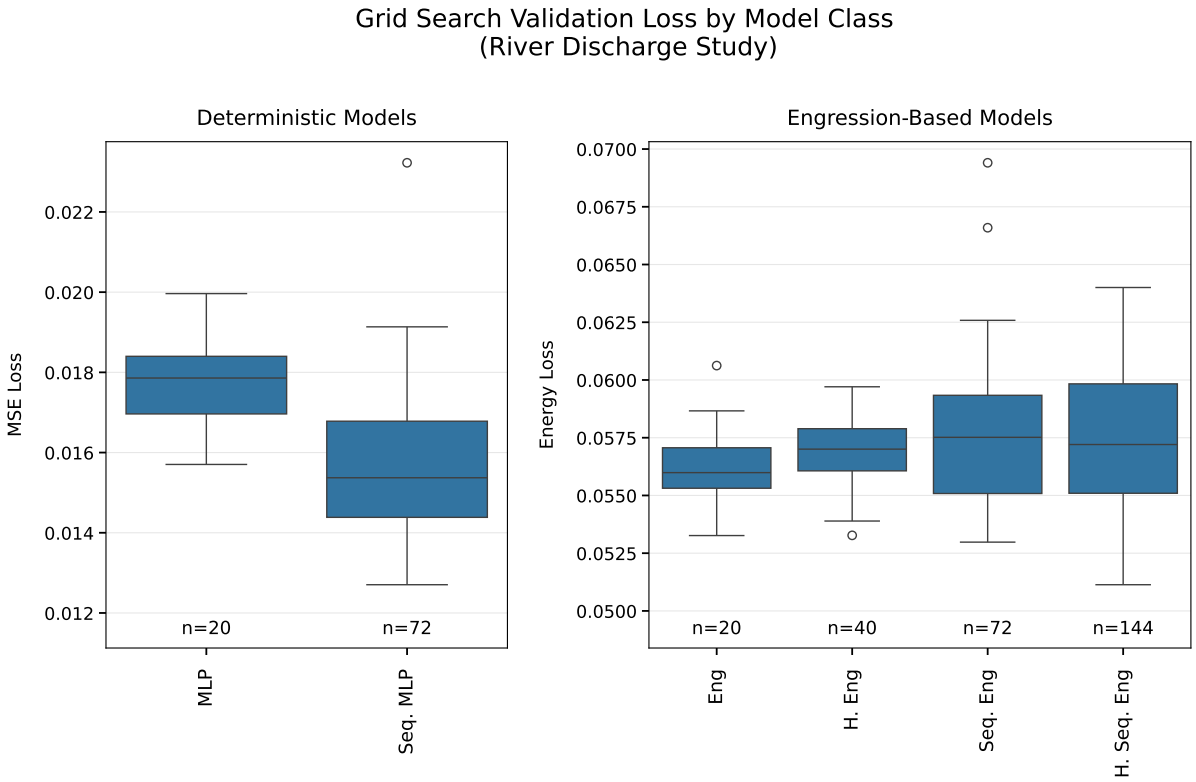

Figure 20: Box plots showing validation loss distributions (on standardized scale) across all hyperparameter combinations tested for each model in the river discharge study.