DEEP ZAKAIJ: STRUCTURED FILTERING FOR JUMP-DIFFUSION TIME SERIES FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Time series driven by unobserved latent states frequently exhibit abrupt jump discontinuities whose timing and magnitude cannot be predicted from observed history alone. Classical jump-diffusion models offer a principled mathematical framework but assume rigid parametric forms, while recent neural jump models operate on fully observed trajectories without inferring the hidden states that govern the dynamics. We propose *Deep ZakaiJ*, a latent-state model for partially observed jump-diffusion systems that embeds the Zakai nonlinear filtering equation into a neural encoder–decoder architecture. The encoder recursively updates a belief over the latent state via Strang splitting into three interpretable substeps, namely prior propagation, diffusion innovation, and jump innovation, yielding a differentiable, first-order-accurate approximation of the exact filtering evolution. The decoder is a structured jump-diffusion model conditioned on the latent state, preserving the separation between continuous dynamics and discontinuous shocks. On synthetic and financial datasets, *Deep ZakaiJ* improves distributional forecasts while remaining competitive in point accuracy, achieving sharper and better-calibrated predictive intervals than the baselines and recovering interpretable latent structure in synthetic and qualitative case studies.

1 INTRODUCTION

Empirical time series rarely evolve smoothly; instead, they are frequently disrupted by abrupt jumps (Aminikhanghahi & Cook, 2017; Short, 2013). These discontinuities span diverse systems, from financial assets suffering sudden price shocks due to macroeconomic news (Merton, 1976), to electricity spot prices spiking drastically when supply runs short (Cartea & Figueroa, 2005; Weron, 2014). Because these extreme dislocations are usually driven by latent states, predicting such trajectories remains a major open challenge.

Classical stochastic models, such as Merton's jump-diffusions (Merton, 1976), stochastic volatility models (Heston, 1993) or more general Lévy processes (Bertoin, 1996), offer a principled mathematical foundation. While models like Heston explicitly introduce a latent volatility state to drive dynamics, they are limited by rigid parametric assumptions and are structurally restricted from capturing complex latent structure. Beyond standard deep learning time-series models (Yu et al., 2019; Benidis et al., 2022; Lim & Zohren, 2021), a growing line of work models sequential data through continuously evolving latent-state dynamics (e.g., Latent ODEs (Rubanova et al., 2019) and Neural CDEs (Kidger et al., 2020)), which propagate hidden states via learned vector fields and approximate the latent posterior through amortized variational inference, but treat latent states as implicit representations rather than explicit stochastic objects, lacking a principled Bayesian update mechanism. Models like CRU (Schirmer et al., 2022) and NCDSSM (Ansari et al., 2023) close this gap by embedding continuous-discrete Kalman filtering within deep sequence models, yet remain restricted to linear-Gaussian dynamics. For time series exhibiting jump discontinuities, where jump times are totally inaccessible stopping times that cannot be predicted from the historical filtration (Protter, 2005; Ankirchner, 2008), correctly updating latent beliefs demands a nonlinear filtering mechanism that current frameworks cannot provide.

In this work, we propose *Deep ZakaiJ*, a framework that embeds the Zakai nonlinear filtering equation into a neural architecture for structured inference in partially observed jump-diffusion systems. Our main contributions are:

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

(i) A Zakai encoder that discretizes the Zakai equation via Strang splitting into three interpretable steps, namely prior propagation, diffusion innovation, and jump innovation. The jump innovation step performs likelihood reweighting against a mixture of no-jump and jump-occurred hypotheses, enabling the posterior to react sharply to dislocations. We prove that this encoder achieves first-order global accuracy relative to the exact Zakai evolution.

(ii) A structured jump-diffusion decoder that parameterizes drift, volatility, jump intensity, and a Gaussian jump-size law with neural networks conditioned on the observation and the latent state, learning the predictable compensator so that the residual reduces to a zero-mean martingale, and coupling forecasting directly to the latent filtering state.

(iii) Experiments on synthetic and financial datasets show that *Deep ZakaiJ* improves distributional quality while remaining competitive in point accuracy, achieving the lowest interval score for the central 90% predictive interval on every dataset and recovering interpretable latent structure in synthetic and qualitative case studies.

2 Related work

Time series forecasting has evolved from classical methods such as ARIMA (Ho & Xie, 1998) and LSTM (Yu et al., 2019) to Transformer variants (e.g., PatchTST (Nie et al., 2023), iTransformer (Liu et al., 2024)), probabilistic models like DeepAR (Salinas et al., 2020), and pretrained foundation models (e.g., Chronos (Ansari et al., 2024), TimesFM (Das et al., 2024), Moirai-MoE (Liu et al., 2025), Lag-Llama (Rasul et al., 2023)). While increasingly powerful, these approaches typically operate in discrete time without explicit mechanisms for jump discontinuities or latent regime inference.

Latent-state models. Early variational approaches such as VRNN (Chung et al., 2015), Deep Kalman Filters (Krishnan et al., 2015), DVBF (Karl et al., 2017), and KVAE (Fraccaro et al., 2017) introduce stochastic latent variables into sequential models but train posteriors via ELBOs rather than filtering equations. SRNN (Fraccaro et al., 2016) stacks a deterministic RNN with a stochastic latent chain and uses a backward pass for smoothing, SNLDS (Dong et al., 2020) collapses discrete switching variables analytically and amortizes the continuous latent posterior to capture regime changes. Deterministic structured SSMs (S4 (Gu et al., 2022), Mamba (Gu & Dao, 2024)) achieve efficient long-range modeling but do not perform stochastic inference. Continuous-time extensions including Latent SDEs (Li et al., 2020) and GRU-ODE-Bayes (De Brouwer et al., 2019) approximate the latent posterior through amortized variational inference, treating latent states as implicit representations. CRU (Schirmer et al., 2022) and NCDSSM (Ansari et al., 2023) go further by embedding exact continuous-discrete Kalman filtering into deep models, demonstrating clear benefits of filtering, but remain restricted to linear-Gaussian dynamics. Particle-filter-based methods such as FIVO (Maddison et al., 2017), VSMC (Naesseth et al., 2018), SIXO (Lawson et al., 2022) provide asymptotically exact Bayesian inference through sequential Monte Carlo, while PF-RNN (Ma et al., 2020) embeds a differentiable particle filter directly into a recurrent cell. Recent extensions handle regime-switching systems (Brady et al., 2024). However, these methods rely on sample-based approximations and do not exploit the continuous structure of filtering equations.

Neural-jump models. Neural Jump SDEs (Jia & Benson, 2019) model temporal point processes via piecewise-continuous latent trajectories where discrete events trigger state-dependent jumps. NJ-ODE (Herrera et al., 2021) learns the conditional expectation of a stochastic process continuously in time, with convergence guarantees, by evolving a hidden state via Neural ODEs (Chen et al., 2018) and applying jump updates at each irregular observation. NeuralMJD (Gao et al., 2025) parameterizes a non-stationary Merton jump-diffusion directly from data, and NJDTPP (Zhang et al., 2024b) unifies intensity modeling through neural jump-diffusion SDEs. While NJ-ODE performs a form of implicit filtering through its jump updates, none of these models embeds an explicit nonlinear filtering equation over a latent state to jointly infer hidden regimes and forecast jump-diffusion dynamics.

Nonlinear filtering. The Zakai equation (Zakai, 1969) provides an unnormalized formulation of the nonlinear filtering problem that is naturally amenable to operator splitting. The classical splitting method (Bensoussan et al., 1990; Gobet et al., 2006; Derchu, 2020) decomposes the Zakai PDE into independent substeps, with recent extensions handling joint diffusion and point-process observations (Zhang et al., 2024a). The energy-based deep splitting method (Bågmark et al., 2023) approximates high-dimensional splitting substeps with neural networks. *Deep ZakaiJ* builds on these numerical

foundations but differs in purpose. Rather than solving a filtering PDE in isolation, it embeds the discretized Zakai equation into an end-to-end architecture, jointly performing nonlinear filtering over a hidden regime and structured conditional jump-diffusion prediction.

3 BACKGROUND

Continuous-time stochastic models are widely used for sequential data. However, standard diffusion processes driven solely by Brownian motion cannot capture the abrupt dislocations frequently observed in practice. A natural extension is the jump-diffusion model (Black & Scholes, 1973; Kou, 2002)

$$\mathrm{d}X_t = \mu(t, X_t)\mathrm{d}t + \sigma(t, X_t)\mathrm{d}W_t + \int_{\mathbb{R}} \gamma(t, z, X_{t^-})\, N(\mathrm{d}t, \mathrm{d}z),$$

where W_t is a Brownian motion, the drift μ describes the local deterministic trend, and the diffusion coefficient σ scales the continuous random fluctuations. $N(\mathrm{d}t, \mathrm{d}z)$ is a Poisson random measure with compensator $\nu(\mathrm{d}z)\mathrm{d}t$. Over a short interval, the Poisson term records whether a jump with mark z occurs, the Lévy measure ν determines the expected jump frequency and mark distribution, and γ maps the jump mark to the realized state displacement. The diffusion term therefore explains continuous variation, while the Poisson integral captures rare but substantial moves (Kou, 2002).

In practical forecasting problems, even the jump-diffusion dynamics of the observed series are rarely autonomous, as they are often driven by regimes, stochastic volatility factors, or other latent mechanisms. This leads naturally to a partially observed coupled jump-diffusion, in which the observed process X_t is driven by an unobserved state Θ_t and forecasting must combine state inference with forward simulation (Kurtz & Ocone, 1988; Bandini et al., 2018),

$$\mathrm{d}X_t = \mu(t, X_t, \Theta_t)\,\mathrm{d}t + \sigma(t, X_t, \Theta_t)\,\mathrm{d}W_t^{X,\mathbb{P}} + \int_{\mathbb{R}} \gamma(t, z, X_{t^-}, \Theta_t)\,\widetilde{N}^{X,\mathbb{P}}(\mathrm{d}t, \mathrm{d}z), \tag{1}$$

$$\mathrm{d}\Theta_t = a(t, \Theta_t)\,\mathrm{d}t + b(t, \Theta_t)\,\mathrm{d}W_t^{\Theta,\mathbb{P}} + \int_{\mathbb{R}} k(t, \Theta_{t^-}, z)\,\widetilde{N}^{\Theta,\mathbb{P}}(\mathrm{d}t, \mathrm{d}z), \tag{2}$$

where the dynamics are written under the physical measure $\mathbb{P}$, and $\widetilde{N}^{X,\mathbb{P}}(\mathrm{d}t, \mathrm{d}z) = N^{X,\mathbb{P}}(\mathrm{d}t, \mathrm{d}z) - \nu^X(\mathrm{d}z)\mathrm{d}t$ and $\widetilde{N}^{\Theta,\mathbb{P}}(\mathrm{d}t, \mathrm{d}z) = N^{\Theta,\mathbb{P}}(\mathrm{d}t, \mathrm{d}z) - \nu^\Theta(\mathrm{d}z)\mathrm{d}t$. The coefficients a, b, and k govern the latent dynamics in a similar manner. Crucially, this formulation preserves the structural advantages of jump-diffusion modeling while operating in the partially observed regime. It thus serves as a powerful counterpart to recent neural jump-process architectures, which fundamentally restrict their scope to fully observed trajectories (Jia & Benson, 2019; Zhang et al., 2024b; Gao et al., 2025). Because Θ_t is not observable, forecasting cannot condition on the latent state directly. The relevant predictive state is instead its conditional law together with a low-dimensional summary statistic,

$$\pi_t(\mathrm{d}\theta) = \mathbb{P}\big(\Theta_t \in \mathrm{d}\theta \mid \mathcal{F}_t^X\big), \qquad \beta_t = \int \varphi(\theta)\,\pi_t(\mathrm{d}\theta),$$

where $\mathcal{F}_t^X = \sigma\{X_s : s \le t\}$ denotes the observation filtration. Here π_t is the nonlinear filter, and β_t is a low-dimensional summary of it defined by a smooth bounded test function φ. For jump-diffusion observations, it is often more convenient to work under a reference measure and study the corresponding unnormalized conditional law (Kurtz & Ocone, 1988; Ceci & Colaneri, 2014).

Proposition 1 (Reference measure and unnormalized filter)**.** *Suppose there exist predictable processes $\eta_0(t) \in \mathbb{R}$ and $\eta_1(t, z) \in (-\infty, 1)$ such that*

$$\sigma(t, X_t, \Theta_t)\,\eta_0(t) + \int_{\mathbb{R}} \gamma(t, z, X_{t^-}, \Theta_t)\,\eta_1(t, z)\,\nu^X(\mathrm{d}z) = \mu(t, X_t, \Theta_t).$$

Then there exists a probability measure $\mathbb{Q}$ equivalent to $\mathbb{P}$, with density process $Z_t = \mathrm{d}\mathbb{Q}/\mathrm{d}\mathbb{P}\,|_{\mathcal{F}_t}$ on the filtration $\mathcal{F}_t$ generated by (X, Θ), under which X_t has no drift, and for any smooth bounded test function φ,

$$\mathbb{E}_{\mathbb{P}}\big[\varphi(\Theta_t) \mid \mathcal{F}_t^X\big] = \frac{q_t(\varphi)}{q_t(1)}, \qquad q_t(\varphi) = \mathbb{E}_{\mathbb{Q}}\big[\varphi(\Theta_t)\,Z_t^{-1} \mid \mathcal{F}_t^X\big].$$

Here η_0 and η_1 are the diffusion and jump likelihood-ratio coefficients, and $Z_t^{-1} = \mathrm{d}\mathbb{P}/\mathrm{d}\mathbb{Q}\,|_{\mathcal{F}_t}$ reweights the paths generated under the driftless reference measure. The filter is therefore recovered

by normalizing q_t, which is the Kallianpur–Striebel formula for jump-diffusion observations. The explicit density process and the proof are deferred to Appendix A.1.

Let $\mathcal{L}_t$ denote the infinitesimal generator of the latent process (2), defined for a smooth bounded test function φ by $(\mathcal{L}_t\varphi)(\theta) = \lim_{h\downarrow 0} h^{-1}\big(\mathbb{E}[\varphi(\Theta_{t+h}) \mid \Theta_t = \theta] - \varphi(\theta)\big)$. It describes the instantaneous evolution of the latent state and collects the drift, diffusion and jump terms of (2); its explicit form is given in Appendix A.2. Writing $q_t(\varphi) = \int \varphi(\theta)\, q_t(\mathrm{d}\theta)$, the unnormalized filter evolves under Proposition 1 in weak form as

$$\mathrm{d}q_t(\varphi) = q_t(\mathcal{L}_t\varphi)\,\mathrm{d}t + q_t(\eta_0\,\varphi)\,\mathrm{d}W_t^{X,\mathbb{Q}} + \int_{\mathbb{R}} q_{t^-}(\hat{\eta}_1(\cdot, z)\,\varphi)\,\widetilde{N}^{X,\mathbb{Q}}(\mathrm{d}t, \mathrm{d}z), \tag{3}$$

where $W^{X,\mathbb{Q}}$ and $\widetilde{N}^{X,\mathbb{Q}}$ are the Brownian motion and the compensated Poisson random measure of the observations under $\mathbb{Q}$, and $\hat{\eta}_1 = \eta_1/(1-\eta_1)$ is the transformed jump coefficient. The first term propagates the unnormalized law along the latent dynamics, while the second and third terms reweight it according to the continuous and jump components of the observations. The derivation is given in Appendix A.2.

Theorem 2 (Existence and uniqueness of the jump-diffusion Zakai equation)**.** *Under standard assumptions for jump-diffusion observations (Ceci & Colaneri, 2014), the Zakai equation* (3) *admits a unique solution in the class of measure-valued processes.*

We therefore work with the unnormalized law q_t rather than the filter π_t, since π_t satisfies a nonlinear equation whereas Equation (3) is linear in q_t. Motivated by this structure, we parameterize the latent belief by an unnormalized density $q_k(\theta) \approx q_{t_k}(\theta)$ on a fixed latent grid, where $q_t(\theta)$ denotes the density of q_t, and its normalized counterpart by $\pi_k(\theta) = q_k(\theta)/\int q_k(\vartheta)\,\mathrm{d}\vartheta$. This parametrization is crucial because the multiplicative structure of the innovation terms makes the Zakai dynamics particularly amenable to operator splitting and differentiable likelihood reweighting.

4 METHODS

Given a past observation window $\{X_{t_0}, \ldots, X_{t_M}\}$, our goal is to infer the current latent belief state and then predict the future path $\{X_{t_{M+1}}, \ldots, X_{t_{M+N}}\}$ over a horizon of N steps. At the observation times $t_0 < t_1 < \cdots$ we write $\Delta X_k = X_{t_{k+1}} - X_{t_k}$ and use the shorthand $\pi_k \equiv \pi_{t_k}$. At a high level, the model consists of a Zakai encoder $\mathcal{E}_\psi$ that recursively updates the unnormalized belief q_k, and a structured jump-diffusion decoder that parameterizes the conditional law of an increment given a latent value:

$$q_{k+1} = \mathcal{E}_\psi(q_k, \Delta X_k; X_{t_k})\,,$$
$$p_\phi(\Delta X_k \mid X_{0:k}) = \int p_\phi(\Delta X_k \mid t_k, X_{t_k}, \theta)\ \pi_k(\theta)\,\mathrm{d}\theta.$$

This encoder–decoder decomposition is the key design principle of our framework. An overview of the full architecture is shown in Figure 1. It retains the uncertainty-aware structure of nonlinear filtering, but replaces analytically intractable operators with differentiable approximations that can be trained end-to-end.

4.1 DEEP ZAKAIJ ENCODER

Our encoder is a discretization of the Zakai equation (3) rather than an ad hoc recurrent state update. Over one interval $[t_k, t_{k+1}]$, the generator term of (3) transports probability mass across latent values, whereas the two innovation terms multiply q_t pointwise in θ by the likelihood of the observed increment ΔX_k, through its continuous and jump components. Each term is easy to apply on its own, as a Markov transition or a Bayes reweighting on the latent grid, but their joint evolution has no closed form. Following splitting-up methods for nonlinear filtering and their learned variants (Bensoussan et al., 1990; Zhang et al., 2024a; Bågmark et al., 2023; Spiteri et al., 2025), we therefore apply them one after another within each step, with $\mathcal{A}_{\Delta t}$ for the generator term, $\mathcal{B}_h$ for the continuous innovation and $\mathcal{C}_h$ for the jump innovation. Applying the three operators once in a fixed order introduces a leading error from their non-commutativity. The symmetric Strang arrangement, which splits each

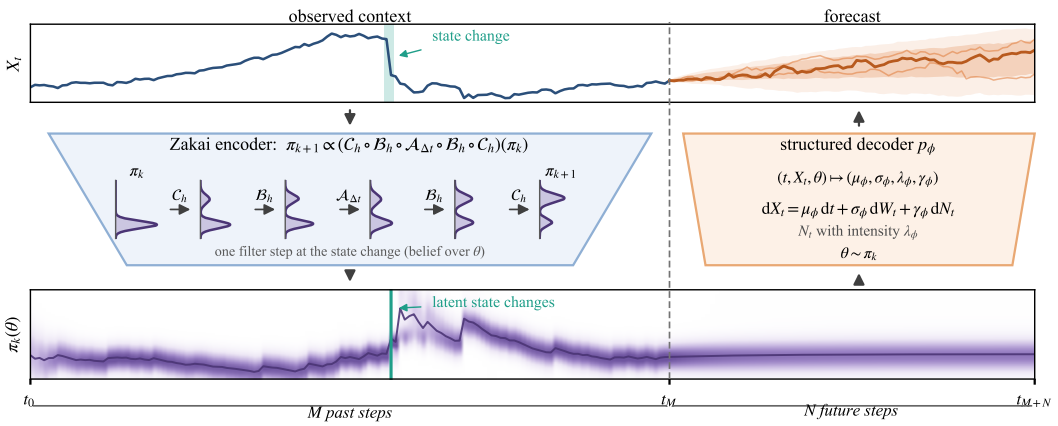


Figure 1: Overview of *Deep ZakaiJ* on a toy example. The Zakai encoder filters the M past steps (middle: one step at the state change; bottom: belief $\pi_k(\theta)$ over time). Over the N future steps the belief evolves by $\mathcal{A}_{\Delta t}$ alone and the structured decoder simulates the forecast (top).

innovation into two half-steps of length $h = \Delta t/2$ placed on either side of $\mathcal{A}_{\Delta t}$, cancels this leading term (Strang, 1968), which is why $\mathcal{B}$ and $\mathcal{C}$ appear at both ends of the update (Appendix A.5)

$$q_{k+1} \approx \mathcal{C}_h \circ \mathcal{B}_h \circ \mathcal{A}_{\Delta t} \circ \mathcal{B}_h \circ \mathcal{C}_h(q_k). \tag{4}$$

$\mathcal{A}$-step: prior propagation. Let $\mathcal{L}_t^*$ denote the forward operator, the adjoint of the generator $\mathcal{L}_t$ of the latent process in (2). The exact prior evolution over one step is $S_{\Delta t}^{\mathcal{A}} = \exp(\Delta t \mathcal{L}_t^*)$, but in general it is not available in closed form. We therefore use a numerical propagator $\widetilde{S}_{\Delta t}^{\mathcal{A}}$ defined on a latent grid $\{\theta_j\}_{j=1}^G$:

$$\widetilde{S}_{\Delta t}^{\mathcal{A}}[q](\theta_j) = \sum_{i=1}^{G} K_{\Delta t}(\theta_j \mid \theta_i)\, q(\theta_i)\, \Delta\theta, \qquad K_{\Delta t}(\cdot \mid \theta_i) \geq 0, \quad \sum_{j=1}^{G} K_{\Delta t}(\theta_j \mid \theta_i) \Delta\theta = 1.$$

where $K_{\Delta t}$ is the transition kernel of a reference latent diffusion, namely the exact kernel of Θ_t when its dynamics are known and a driftless Brownian kernel otherwise (Appendix B.2). To account for numerical bias and model mismatch, we add a learned residual correction $R_\psi^{\mathcal{A}}(q; \Delta t)$ with zero total mass. The $\mathcal{A}$-step is $\mathcal{A}_{\Delta t}[q](\theta_j) = \left[\widetilde{S}_{\Delta t}^{\mathcal{A}}[q](\theta_j) + R_\psi^{\mathcal{A}}(q; \Delta t)(\theta_j)\right]^+$, where $[\cdot]^+ = \max(\cdot, 0)$ clips negative values. This preserves the physical prior while allowing the encoder to learn systematic corrections that are useful for prediction.

$\mathcal{B}$-halfstep: diffusion innovation. Conditioned on a candidate latent value θ, the continuous part of the observed increment is approximated by $\Delta X_k \mid \theta \approx \mathcal{N}(\mu\Delta t, \sigma^2 \Delta t)$, with the coefficients evaluated at (t_k, X_{t_k}, θ). This defines the continuous likelihood factor $\ell_{\mathcal{B},k}(\theta_j) = \mathcal{N}_0(\theta_j)$, where $\mathcal{N}_0 = \mathcal{N}(\Delta X_k; \mu\Delta t, \sigma^2 \Delta t)$. Reweighting by this factor sharpens the posterior in regions of latent space that explain the continuous component of the observed move.

$\mathcal{C}$-halfstep: jump innovation. Given θ_j, jumps of X arrive with intensity $\lambda(t_k, X_{t_k}, \theta_j)$ and have Gaussian sizes with mean $\gamma(t_k, X_{t_k}, \theta_j)$ and standard deviation $\delta(t_k, X_{t_k}, \theta_j)$, as in the Merton model (Merton, 1976). For small Δt we neglect two or more jumps in one step. Given one jump, the increment is Gaussian with mean $\mu\Delta t + \gamma$ and variance $\sigma^2\Delta t + \delta^2$, the sum of the diffusion and jump-size variances, with density $\mathcal{N}_1 = \mathcal{N}(\Delta X_k; \mu\Delta t + \gamma, \sigma^2\Delta t + \delta^2)$. Hence the likelihood of ΔX_k is $\ell_k = e^{-\lambda\Delta t}(\mathcal{N}_0 + \lambda\Delta t\, \mathcal{N}_1) = \ell_{\mathcal{B},k}\, \ell_{\mathcal{C},k}$, with the jump factor

$$\ell_{\mathcal{C},k}(\theta_j) = e^{-\lambda\Delta t}\left(1 + \lambda\Delta t\, \frac{\mathcal{N}_1}{\mathcal{N}_0}\right), \tag{5}$$

where the dependence of μ, σ, λ, γ and δ on (t_k, X_{t_k}, θ_j) is suppressed. The factor $\ell_{\mathcal{C},k}$ compares the jump and no-jump explanations of the observed increment. Each half-step multiplies the belief by half of its factor,

$$\mathcal{B}_h[q](\theta_j) = q(\theta_j)\, \ell_{\mathcal{B},k}(\theta_j)^{1/2}, \qquad \mathcal{C}_h[q](\theta_j) = q(\theta_j)\, \ell_{\mathcal{C},k}(\theta_j)^{1/2}.$$

Since $\mathcal{B}$ and $\mathcal{C}$ are pointwise multiplications, they commute, and $\mathcal{C}_h \circ \mathcal{B}_h$ multiplies q by $\ell_k^{1/2}$ with $\ell_k = \ell_{\mathcal{B},k}\,\ell_{\mathcal{C},k}$, so (4) evaluates the likelihood of ΔX_k symmetrically at both ends of the step. This construction lets the posterior react much more strongly to large observed dislocations than a purely Gaussian filter would, while keeping the update differentiable and numerically stable.

4.2 Structured decoder and joint training

The decoder specifies the conditional one-step law of future increments given the current observation and a candidate latent state. We constrain this law to a jump-diffusion family rather than an unconstrained transition density, which separates continuous volatility from jump risk and lets the same coefficients serve as the likelihood in the encoder and as the simulator in forecasting. Unlike earlier neural jump models (Jia & Benson, 2019; Gao et al., 2025; Zhang et al., 2024b), which act on fully observed trajectories, the decoder is coupled to a filtered latent state.

The decoder parameterizes the coefficients of Section 4.1 as $\mu_\phi(t,x,\theta)$, $\sigma_\phi(t,x,\theta)$, $\lambda_\phi(t,x,\theta)$, $\gamma_\phi(t,x,\theta)$ and $\delta_\phi(t,x,\theta)$ with a compact neural network. Positivity is imposed on the volatility, the jump intensity and the jump-size dispersion.

A Lévy measure may have infinite mass near zero, in which case the process has infinitely many jumps in every time interval, almost all of them arbitrarily small. At a fixed sampling frequency, these small jumps cannot be distinguished from diffusive fluctuations in the observed increments. The following result makes this precise for Lévy processes.

Theorem 3 (Small-jump Gaussian approximation (Asmussen & Rosiński, 2001)). *Let J be a Lévy process with Lévy measure ν, and for $\varepsilon > 0$ let $J^{(\varepsilon)}$ denote its compensated component of jumps with magnitude at most ε, with variance per unit time $s^2(\varepsilon) = \int_{|z|\le\varepsilon} z^2\,\nu(\mathrm{d}z)$. If $s(\varepsilon)/\varepsilon \to \infty$ as $\varepsilon \to 0$, then $s(\varepsilon)^{-1} J^{(\varepsilon)}$ converges in distribution to a standard Brownian motion.*

By Theorem 3, jumps below ε contribute approximately a Brownian term with variance $s^2(\varepsilon)$ per unit time. We therefore interpret σ_ϕ as an effective volatility that includes this contribution, and only jumps larger than ε enter the jump component $(\lambda_\phi, \gamma_\phi, \delta_\phi)$ of the $\mathcal{C}$-step.

Forecasting. During filtering over the observed history, the encoder applies the full $\mathcal{A}/\mathcal{B}/\mathcal{C}$ update. Beyond the last observation there is no increment to reweight by, so the belief is propagated by the $\mathcal{A}$-step alone. Each forecast trajectory draws θ from the current belief, simulates one step of the decoder's jump-diffusion by the Euler–Maruyama method, and propagates the belief with $\mathcal{A}_{\Delta t}$ (Algorithm B.3). The predictive uncertainty therefore widens as the belief spreads under the latent prior.

Training. Training uses sliding windows with a past context segment $X_{s:s+M}$ and a future segment $X_{s+M:s+M+N}$ whose increments serve as forecasting targets. Given the context, the encoder produces the latent posteriors, and we maximize the stepwise filtering-and-forecasting objective

$$\mathcal{L}(\phi,\psi) = \sum_{k=0}^{M+N-1} \mathbb{E}_{\pi_k(\theta)}[\log p_\phi(\Delta X_k \mid t_k, X_{t_k}, \theta)] - \sum_{k=1}^{M} \mathrm{KL}\Big(\pi_k \,\Big\|\, \pi_k^{\mathrm{prior}}\Big)\,,$$

where $\pi_k^{\mathrm{prior}} \propto \widetilde{S}^{\mathcal{A}}_{\Delta t}[u_k]$ is the output of the reference kernel, before the learned residual, applied to the density $u_k = \mathcal{B}_h \circ \mathcal{C}_h(q_{k-1})$ that enters the $\mathcal{A}$-step. Gradients flow through all splitting operations and normalizations, so that the encoder and decoder are trained end to end.

4.3 Approximation error of the split encoder

Proposition 1 and Theorem 2 establish the continuous-time filtering model, and Theorem 3 justifies the treatment of small jumps; it remains to bound the error introduced by the split encoder itself.

Assumption 4 (L^1-stability). *There exists $L \ge 0$ such that for any normalized densities q_1 and q_2,*

$$\|\mathcal{S}_{\Delta t}(q_1) - \mathcal{S}_{\Delta t}(q_2)\|_1 \le (1 + L\Delta t)\,\|q_1 - q_2\|_1\,,$$

where $\mathcal{S}_{\Delta t}$ denotes the exact one-step filtering semigroup.

Theorem 5 (Local approximation error of the split encoder). *Let $\widetilde{\mathcal{S}}_{\Delta t}$ denote the update* (4) *with the at-most-one-jump likelihood* (5) *and the exact prior propagator $S^{\mathcal{A}}_{\Delta t} = \exp(\Delta t\,\mathcal{L}_t^*)$ in the $\mathcal{A}$-step.*

Assume Assumption 4 and suppose the intensity of jumps larger than ε is uniformly bounded by Λ_ε. Then there exist constants $C_1, C_2 > 0$, independent of Δt, such that for any normalized input density q,

$$\left\| \widetilde{\mathcal{S}}_{\Delta t}(q) - \mathcal{S}_{\Delta t}(q) \right\|_1 \leq C_1 \Delta t^2 + C_2 (\Lambda_\varepsilon \Delta t)^2.$$

The first term is the splitting error (Bensoussan et al., 1990; Strang, 1968), and the second arises from neglecting two or more jumps in one step, controlled by $\mathbb{P}(N_{\Delta t} \geq 2) = \mathcal{O}((\Lambda_\varepsilon \Delta t)^2)$. Accumulating the local error over a fixed horizon gives the global rate.

Corollary 6 (Global error over a fixed horizon)**.** *Fix $T > 0$ and let $n = T/\Delta t$. Under the assumptions of Theorem 5, there exists a constant $C_T > 0$, independent of Δt, such that*

$$\left\| \widetilde{\mathcal{S}}^n_{\Delta t}(q) - \mathcal{S}^n_{\Delta t}(q) \right\|_1 \leq C_T (1 + \Lambda_\varepsilon^2) \Delta t.$$

Thus, the split encoder is first-order accurate over a fixed horizon. Since the encoder reads out the normalized belief π_k, the following proposition transfers the bound from q to π.

Proposition 7 (Normalization stability)**.** *Let $q, \tilde{q}$ be nonnegative integrable functions with $\|q\|_1 > 0$. Then*

$$\left\| \frac{\tilde{q}}{\|\tilde{q}\|_1} - \frac{q}{\|q\|_1} \right\|_1 \leq \frac{2}{\|q\|_1} \|\tilde{q} - q\|_1 .$$

Combined with Theorem 5 and Corollary 6, the normalized encoder is first-order accurate whenever the unnormalized mass $\|q\|_1$ stays bounded away from zero. The effects of the grid kernel $K_{\Delta t}$ and the learned residual $R^{\mathcal{A}}_\psi$ are measured in Appendices C.3 and C.4. Full proofs are provided in Appendix A.

5 Experiments

We evaluate *Deep ZakaiJ* on one synthetic benchmark and two real-world datasets, covering both controlled settings and practical forecasting scenarios; a third real series, MSTR perpetual futures, is reported in Appendix C.2. Our goal is to assess not only predictive accuracy, but also the ability to capture stochastic dynamics and jump behaviors. Implementation details, preprocessing, and hyperparameters are deferred to Appendix B, and ablations of each component are reported in Appendix C.3.

Baselines. We compare *Deep ZakaiJ* with a diverse set of models spanning different modeling paradigms. First, we include DLinear (Zeng et al., 2023) as a simple linear reference and classical sequence models LSTM (Yu et al., 2019) and DeepAR (Salinas et al., 2020). Transformer-based forecasting is covered by PatchTST (Nie et al., 2023). To benchmark latent-state modeling, we include LatentODE (Rubanova et al., 2019), LatentSDE (Li et al., 2020) and NCDSSM (Ansari et al., 2023); these models perform latent-state inference but are restricted to continuous or linear-Gaussian dynamics without explicit jump mechanisms. For neural jump-diffusion modeling, we compare against NJ-ODE (Herrera et al., 2021) and NeuralMJD (Gao et al., 2025), which capture jump dynamics but operate on fully observed trajectories without latent-state filtering. Finally, we include Chronos (Ansari et al., 2024) as a representative foundation model. Additional baselines are reported in Appendix C.1.

Evaluation metrics. We evaluate both point and probabilistic forecasts. For point accuracy, we report MAE and RMSE. For probabilistic quality, we use CRPS (Gneiting & Raftery, 2007) and average log-likelihood, which assess distributional sharpness and calibration. We also report the interval score of the central 90% predictive interval (IS90) (Winkler, 1972; Gneiting & Raftery, 2007). With $\alpha = 0.1$ and the interval $[l, u] = [\mu - z_{0.95}\sigma,\ \mu + z_{0.95}\sigma]$ formed from the predictive mean and standard deviation, the score of an observation y is $(u - l) + \frac{2}{\alpha}(l - y)\mathbf{1}\{y < l\} + \frac{2}{\alpha}(y - u)\mathbf{1}\{y > u\}$, which rewards narrow intervals and penalizes observations outside them in proportion to the miss distance. Lower values indicate better performance for MAE, RMSE, CRPS and IS90, while higher log-likelihood is preferred.

Table 1: Results on the synthetic dataset, mean ± standard deviation over five seeds.

Model	MAE	RMSE	CRPS	LogLik	IS90
DLinear	0.4389±.0195	0.5038±.0236	–	–	–
LSTM	0.1192±.0055	0.1681±.0065	–	–	–
DeepAR	3.3361±1.9932	3.4207±2.0498	2.6306±1.7143	−4.62±2.56	30.296±25.052
PatchTST	0.1043±.0027	0.1477±.0045	–	–	–
LatentSDE	0.1027±.0101	0.1453±.0058	0.0910±.0019	−0.38±.05	1.033±.015
LatentODE	0.1385±.0062	0.1871±.0076	0.1030±.0033	−0.51±.09	0.926±.017
NCDSSM	0.1104±.0028	0.1514±.0036	0.0863±.0016	−1.36±.03	0.876±.017
NJ-ODE	0.1184±.0199	0.1668±.0215	–	–	–
NeuralMJD	0.0895±.0053	0.1384±.0104	0.0690±.0052	−1.20±2.66	0.786±.289
Chronos	0.1383±.0013	0.1855±.0021	0.2197±.0609	−7.1e11±7.7e11	2.716±.849
Deep ZakaiJ	**0.0850**±.0009	**0.1367**±.0017	**0.0671**±.0007	**0.73**±.04	**0.620**±.025

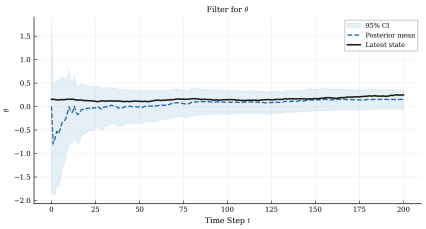


Figure 2: Posterior mean and 95% band of the filtered belief against the true latent state on the synthetic dataset.

5.1 Synthetic Data

Data generation. We evaluate *Deep ZakaiJ* on a synthetic partially observed jump-diffusion process, where both the observation drift and jump intensity depend linearly on the latent state. We simulate a trajectory of 30,000 time steps and construct samples using a sliding window. Conditioning on the past 300 observations, we predict the next 100 steps with stride 100. The data is split into 60% training, 20% validation, and 20% testing. Full details are discussed in Appendix B.1.

Results. Since the data-generating mechanism is known, this synthetic setting tests *Deep ZakaiJ* under a correctly specified linear decoder; polynomial and neural decoders are compared in Appendix C.4. Table 1 shows that *Deep ZakaiJ* achieves the best performance on every metric, including the interval score IS90. Figure 2 further shows that the inferred filter tracks the latent state after a short initial transient. These results suggest that when the decoder is correctly specified, explicit filtering improves both state estimation and downstream forecasting.

5.2 Real-world Data

Gold dataset (XAU/USD). This dataset comprises historical quotes for Gold against the U.S. Dollar, sourced from the Titan FX Research Hub Titan FX (2025). To capture representative price movements, we resample the original one-minute OHLCV records into 10-minute intervals, focusing on the closing prices. We adopt a sliding window configuration with a lookback capacity of 300 ($M = 300$) time steps and a prediction horizon of 100 ($N = 100$) steps, utilizing a stride of 100 for sample generation. To prevent temporal leakage, the data is partitioned chronologically into training (Jan. 2–Aug. 22, 2025), validation (Aug. 22–Oct. 10, 2025), and testing (Oct. 10–Nov. 28, 2025).

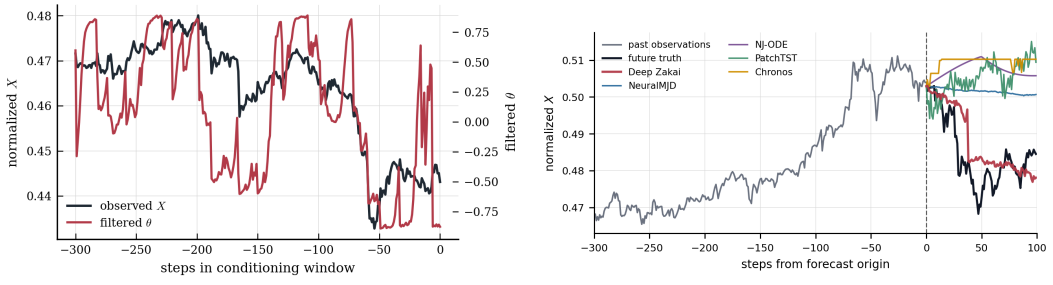
Cryptocurrency dataset (BTC/USDT). This dataset comprises the BTCUSDT perpetual futures contract on Binance (Binance, 2025), which trades around the clock. We use the closing prices of 15-minute bars from January 1 to December 31, 2025, giving 35,040 observations. At a daily scale the series is about twice as volatile as gold, and its 15-minute increments are heavy-tailed with clustered volatility, so that large moves arrive in bursts rather than in isolation. Consistent with the XAU/USD dataset, we implement a sliding window approach ($M = 300, N = 100$) with a stride of 100. The dataset is split into training (Jan. 1–Sep. 13, 2025), validation (Sep. 13–Nov. 7, 2025), and testing (Nov. 7–Dec. 31, 2025) sets, strictly following the chronological order of observations.

Results. As shown in Table 2, *Deep ZakaiJ* achieves the best performance on both datasets across all point and distributional metrics. On XAU/USD, it reduces IS90 from 0.0532 for LatentSDE to 0.0433. On BTC/USDT, it improves CRPS from 0.0085 to 0.0081, LogLik from 2.83 to 2.87 and IS90 from 0.0756 to 0.0737 over NeuralMJD, which models jumps on the observed trajectory without a latent state and is the strongest baseline in both MAE and CRPS on both datasets. Relative to it, *Deep ZakaiJ* reduces MAE by 1.8% and 3.0% but CRPS by 7.9% and 5.2% on XAU/USD and BTC/USDT, so the improvement is concentrated in the predictive distribution. Although Chronos yields competitive point errors on both datasets, its substantially worse LogLik indicates weaker uncertainty calibration.

Regime detection. We further examine whether *Deep ZakaiJ* learns latent structure aligned with plausible financial regimes. As shown in Figure 3a, the filtered belief θ co-evolves with the XAU/USD price but is not a simple echo. It shifts ahead of several trend reversals and reacts sharply to the

Table 2: Results on the two real-world datasets, mean ± standard deviation over five seeds.

	XAU/USD					BTC/USDT				
Model	MAE	RMSE	CRPS	LogLik	IS90	MAE	RMSE	CRPS	LogLik	IS90
DLinear	0.0119±.0004	0.0153±.0004	–	–	–	0.0197±.0007	0.0261±.0008	–	–	–
LSTM	0.0081±.0006	0.0116±.0013	–	–	–	0.0114±.0005	0.0166±.0006	–	–	–
DeepAR	0.0878±.0706	0.0983±.0768	0.0708±.0609	−0.93±2.87	0.8704±.9003	0.0282±.0060	0.0348±.0061	0.0204±.0042	1.73±.35	0.1703±.0393
PatchTST	0.0100±.0002	0.0133±.0002	–	–	–	0.0172±.0010	0.0234±.0012	–	–	–
LatentSDE	0.0083±.0002	0.0114±.0002	0.0061±.0001	3.03±.02	0.0532±.0005	0.0117±.0003	0.0172±.0001	0.0093±.0001	2.14±.02	0.0948±.0003
LatentODE	0.0126±.0001	0.0171±.0001	0.0093±.0000	2.34±.01	0.0797±.0003	0.0191±.0005	0.0256±.0007	0.0139±.0004	1.64±.05	0.1244±.0030
NCDSSM	0.0082±.0001	0.0114±.0001	0.0062±.0000	2.52±.04	0.0588±.0006	0.0117±.0000	0.0174±.0001	0.0093±.0000	1.30±.05	0.0991±.0011
NJ-ODE	0.0084±.0001	0.0114±.0001	–	–	–	0.0108±.0001	0.0167±.0002	–	–	–
NeuralMJD	0.0076±.0003	0.0109±.0002	0.0058±.0002	2.71±.04	0.0583±.0016	0.0108±.0002	0.0164±.0002	0.0085±.0002	2.83±.03	0.0756±.0036
Chronos	0.0079±.0001	0.0112±.0002	0.0074±.0011	−4.7e4±1.1e5	0.0863±.0153	0.0114±.0002	0.0169±.0002	0.0130±.0031	−1.1e4±2.3e4	0.1550±.0444
Deep ZakaiJ	**0.0075**±.0001	**0.0107**±.0001	**0.0054**±.0001	**3.32**±.01	**0.0433**±.0004	**0.0105**±.0001	**0.0158**±.0001	**0.0081**±.0001	**2.87**±.01	**0.0737**±.0014



(a) XAU/USD conditioning window.

(b) XAU/USD 100-step forecast.

Figure 3: Qualitative examples on XAU/USD. (a) The filtered θ changes state within the observed conditioning window. (b) Conditioned on the past 300 observations, *Deep ZakaiJ* and selected baselines are compared over the next 100 steps.

dislocation after step -50, where the jump innovation concentrates posterior mass on a low regime. The forecast in Figure 3b suggests that this helps *Deep ZakaiJ* adapt after the transition. If the belief tracks regimes, its change points should coincide with shifts in volatility. We detect change points separately in the filtered posterior mean and in a rolling log-volatility series and match them within 25 steps, which gives F1 0.78 ± 0.02 on XAU/USD and 0.43 ± 0.07 on BTC/USDT, about twice the values of 0.42 and 0.23 obtained after circularly shifting the belief change points. At the belief change points, the absolute rank correlation between the change in the belief and the change in log-volatility exceeds 0.86 on both datasets (details in Appendix C.6).

6 Conclusion

We introduced *Deep ZakaiJ*, a latent-state model for partially observed jump-diffusion systems that embeds the Zakai nonlinear filtering equation into an end-to-end trainable neural architecture. By maintaining an explicit belief over the hidden regime through principled filtering, the model separates latent-state inference from structured forecasting, learns latent representations that are interpretable in synthetic and qualitative case studies, keeps prior propagation, diffusion innovation and jump innovation as separate operators with a first-order error bound for the split encoder, and produces predictive intervals with the lowest interval score among the evaluated models. Experiments show that this filtering-driven approach performs strongly among the evaluated baselines, particularly in uncertainty quantification. On the real datasets, change points of the filtered belief align with shifts in realized volatility, which supports reading the belief as a regime indicator. Future work includes extending to high-dimensional latent states and multivariate observations, and incorporating exogenous covariates into the belief update.

AI USE STATEMENT

In this work, we used generative AI tools to brainstorm and discuss research ideas, to help organize drafts and compile experimental results into tables, and to review and polish the wording of the completed paper to improve readability. We have not used generative AI tools to generate the main technical contributions or to write the proofs. All tables were checked manually against the underlying experimental outputs, and all AI-assisted text was reviewed and edited by the authors. We take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work does not involve human subjects or personal data. The Binance BTCUSDT and MSTRUSDT perpetual-futures data are publicly available, and the XAU/USD data are used for analysis under the Titan FX Historical Data Terms of Use, which prohibit redistribution, so we cite the source and describe the preprocessing without redistributing the records (Appendix B.1). Potential applications and risks, including overreliance on predictive intervals in high-stakes financial and energy settings, are discussed in Appendix E.

REPRODUCIBILITY STATEMENT

The assumptions and complete proofs of all theoretical results are given in Appendix A. Appendix B.1 describes the data sources, the synthetic data-generating process with all parameter values, and the preprocessing of the real datasets. Appendix B.2 specifies the chronological splits, the forecasting windows, the random seeds, the architecture, the optimization settings and the evaluation protocol, and Algorithms B.2 and B.3 give the training and inference procedures. The source code will be made publicly available upon acceptance.

REFERENCES

Samaneh Aminikhanghahi and Diane J Cook. A survey of methods for time series change point detection. *Knowledge and Information Systems*, 51(2):339–367, 2017.

Stefan Ankirchner. On filtration enlargements and purely discontinuous martingales. *Stochastic Processes and their Applications*, 118(9):1662–1678, 2008.

Abdul Fatir Ansari, Alvin Heng, Andre Lim, and Harold Soh. Neural continuous-discrete state space models for irregularly-sampled time series. In *International Conference on Machine Learning*, 2023.

Abdul Fatir Ansari, Lorenzo Stella, Ali Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Bernie Wang. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=gerNCVqqtR. Expert Certification.

Søren Asmussen and Jan Rosiński. Approximations of small jumps of Lévy processes with a view towards simulation. *Journal of Applied Probability*, 38(2):482–493, 2001.

Kasper Bågmark, Adam Andersson, and Stig Larsson. An energy-based deep splitting method for the nonlinear filtering problem. *Partial Differential Equations and Applications*, 4(2):14, 2023.

Elena Bandini, Andrea Cosso, Marco Fuhrman, and Huyên Pham. Backward SDEs for optimal control of partially observed path-dependent stochastic systems: A control randomization approach. *The Annals of Applied Probability*, 28(3):1634–1678, 2018.

Konstantinos Benidis, Syama Sundar Rangapuram, Valentin Flunkert, Yuyang Wang, Danielle Maddix, Caner Turkmen, Jan Gasthaus, Michael Bohlke-Schneider, David Salinas, Lorenzo Stella, et al. Deep learning for time series forecasting: Tutorial and literature survey. *ACM Computing Surveys*, 55(6):1–36, 2022.

Allan Bensoussan, Roland Glowinski, and Aurel Rascanu. Approximation of the Zakai equation by the splitting up method. *SIAM Journal on Control and Optimization*, 28(6):1420–1431, 1990.

Jean Bertoin. *Lévy Processes*, volume 121 of *Cambridge Tracts in Mathematics*. Cambridge University Press, Cambridge, 1996.

Binance. Binance Public Data: BTCUSDT USD-M Perpetual Futures Klines. `https://data.binance.vision/?prefix=data/futures/um/monthly/klines/BTCUSDT/15m/`, 2025. Fifteen-minute klines, January–December 2025.

Binance. Binance Public Data: MSTRUSDT USD-M Perpetual Futures Klines. `https://data.binance.vision/?prefix=data/futures/um/monthly/klines/MSTRUSDT/1m/`, 2026. One-minute klines, February–August 2026.

Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–654, 1973.

John-Joseph Brady, Yuhui Luo, Wenwu Wang, Víctor Elvira, and Yunpeng Li. Regime learning for differentiable particle filters. In *2024 27th International Conference on Information Fusion (FUSION)*, pp. 1–6. IEEE, 2024.

Álvaro Cartea and Marcelo G. Figueroa. Pricing in electricity markets: a mean reverting jump diffusion model with seasonality. *Applied Mathematical Finance*, 12(4):313–335, 2005. doi: 10.1080/13504860500117503.

Claudia Ceci and Katia Colaneri. The Zakai equation of nonlinear filtering for jump-diffusion observations: Existence and uniqueness. *Applied Mathematics & Optimization*, 69(1):47–82, 2014.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Forty-first International Conference on Machine Learning*, 2024. URL `https://openreview.net/forum?id=jn2iTJas6h`.

Edward De Brouwer, Jaak Simm, Adam Arany, and Yves Moreau. GRU-ODE-Bayes: Continuous modeling of sporadically-observed time series. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Joffrey Derchu. A Bayesian viewpoint on the price formation process. *Market Microstructure and Liquidity*, 6(01n04):2050010, 2020.

Zhe Dong, Bryan Seybold, Kevin Murphy, and Hung Bui. Collapsed amortized variational inference for switching nonlinear dynamical systems. In *International Conference on Machine Learning*, pp. 2638–2647. PMLR, 2020.

Marco Fraccaro, Søren Kaae Sønderby, Ulrich Paquet, and Ole Winther. Sequential neural models with stochastic layers. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Marco Fraccaro, Simon Kamronn, Ulrich Paquet, and Ole Winther. A disentangled recognition and nonlinear dynamics model for unsupervised learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Yuanpei Gao, Qi Yan, Yan Leng, and Renjie Liao. Neural MJD: Neural non-stationary Merton jump diffusion for time series prediction. *arXiv preprint arXiv:2506.04542*, 2025.

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

Emmanuel Gobet, Gilles Pages, Huyên Pham, and Jacques Printems. Discretization and simulation of the Zakai equation. *SIAM Journal on Numerical Analysis*, 44(6):2505–2538, 2006.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=tEYskw1VY2.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *The International Conference on Learning Representations (ICLR)*, 2022.

Calypso Herrera, Florian Krach, and Josef Teichmann. Neural jump ordinary differential equations: Consistent continuous-time prediction and filtering. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=JFKR3WqwyXR.

Steven L Heston. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *The Review of Financial Studies*, 6(2):327–343, 1993.

Siu Lau Ho and Min Xie. The use of ARIMA models for reliability forecasting and analysis. *Computers & Industrial Engineering*, 35(1-2):213–216, 1998.

Junteng Jia and Austin R. Benson. Neural jump stochastic differential equations. In *Advances in Neural Information Processing Systems*, volume 32, pp. 9843–9854, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/59b1deff341edb0b76ace57820cef237-Abstract.html.

Maximilian Karl, Maximilian Soelch, Justin Bayer, and Patrick van der Smagt. Deep variational Bayes filters: Unsupervised learning of state space models from raw data. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=HyTqHL5xg.

Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural controlled differential equations for irregular time series. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6696–6707, 2020.

Steven G Kou. A jump-diffusion model for option pricing. *Management Science*, 48(8):1086–1101, 2002.

Rahul G Krishnan, Uri Shalit, and David Sontag. Deep Kalman filters. *arXiv preprint arXiv:1511.05121*, 2015.

Thomas G Kurtz and Daniel L Ocone. Unique characterization of conditional distributions in nonlinear filtering. *The Annals of Probability*, pp. 80–107, 1988.

Dieterich Lawson, Allan Raventos, Andrew Warrington, and Scott Linderman. SIXO: Smoothing inference with twisted objectives. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=bDyLgfvZ0qJ.

Dominique Lépingle and Jean Mémin. Sur l'intégrabilité uniforme des martingales exponentielles. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 42(3):175–203, 1978.

Xuechen Li, Ting-Kam Leonard Wong, Ricky TQ Chen, and David Duvenaud. Scalable gradients for stochastic differential equations. In *International Conference on Artificial Intelligence and Statistics*, pp. 3870–3882. PMLR, 2020.

Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379(2194), 2021.

Chenghao Liu, Taha Aksu, Juncheng Liu, Xu Liu, Hanshu Yan, Quang Pham, Silvio Savarese, Doyen Sahoo, Caiming Xiong, and Junnan Li. Moirai 2.0: When less is more for time series forecasting. *arXiv preprint arXiv:2511.11698*, 2025.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=JePfAI8fah.

Xiao Ma, Peter Karkus, David Hsu, and Wee Sun Lee. Particle filter recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 5101–5108, 2020.

Chris J Maddison, John Lawson, George Tucker, Nicolas Heess, Mohammad Norouzi, Andriy Mnih, Arnaud Doucet, and Yee Teh. Filtering variational objectives. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Robert C Merton. Option pricing when underlying stock returns are discontinuous. *Journal of Financial Economics*, 3(1-2):125–144, 1976.

Christian Naesseth, Scott Linderman, Rajesh Ranganath, and David Blei. Variational sequential Monte Carlo. In *International Conference on Artificial Intelligence and Statistics*, pp. 968–977. PMLR, 2018.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with Transformers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Jbdc0vTOcol.

Philip E. Protter. Stochastic differential equations. In *Stochastic Integration and Differential Equations*, Stochastic Modelling and Applied Probability, pp. 249–361. Springer, Berlin, Heidelberg, 2005. doi: 10.1007/978-3-662-10061-5_6.

Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Arian Khorasani, George Adamopoulos, Rishika Bhagwatkar, Marin Biloš, Hena Ghonia, Nadhir Hassen, Anderson Schneider, Sahil Garg, Alexandre Drouin, Nicolas Chapados, Yuriy Nevmyvaka, and Irina Rish. Lag-Llama: Towards foundation models for time series forecasting. In *R0-FoMo:Robustness of Few-shot and Zero-shot Learning in Large Foundation Models*, 2023. URL https://openreview.net/forum?id=jYluzCLFDM.

Yulia Rubanova, Ricky TQ Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3): 1181–1191, 2020.

Mona Schirmer, Mazin Eltayeb, Stefan Lessmann, and Maja Rudolph. Modeling irregular time series with continuous recurrent units. In *International Conference on Machine Learning*, pp. 19388–19405. PMLR, 2022.

Michael Short. Improved inequalities for the Poisson and binomial distribution and upper tail quantile functions. *International Scholarly Research Notices*, 2013(1):412958, 2013.

Raymond J Spiteri, Arash Tavassoli, Siqi Wei, and Andrei Smolyakov. Beyond Strang: A practical assessment of some second-order 3-splitting methods. *Communications on Applied Mathematics and Computation*, 7(1):95–114, 2025.

Gilbert Strang. On the construction and comparison of difference schemes. *SIAM Journal on Numerical Analysis*, 5(3):506–517, 1968.

Titan FX. Titan FX Research Hub: XAUUSD Historical Market Data. https://research.titanfx.com/instruments/xauusd, 2025. One-minute OHLCV quote data for XAUUSD, Standard series.

Rafał Weron. Electricity price forecasting: A review of the state-of-the-art with a look into the future. *International Journal of Forecasting*, 30(4):1030–1081, 2014. doi: 10.1016/j.ijforecast.2014.08.008.

Robert L. Winkler. A decision-theoretic approach to interval estimation. *Journal of the American Statistical Association*, 67(337):187–191, 1972.

Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A review of recurrent neural networks: LSTM cells and network architectures. *Neural Computation*, 31(7):1235–1270, 2019.

Moshe Zakai. On the optimal filtering of diffusion processes. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 11(3):230–243, 1969.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are Transformers effective for time series forecasting? In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

Fengshan Zhang, Yongkui Zou, Shimin Chai, and Yanzhao Cao. A splitting method for nonlinear filtering problems with diffusive and point process observations. *Communications in Computational Physics*, 36(4):996–1020, 2024a.

Shuai Zhang, Chuan Zhou, Yang Aron Liu, Peng Zhang, Xixun Lin, and Zhi-Ming Ma. Neural jump-diffusion temporal point processes. In *Forty-first International Conference on Machine Learning*, 2024b. URL https://openreview.net/forum?id=d1P6GtRzuV.

A PROOFS AND AUXILIARY RESULTS

This appendix contains the proofs and derivations omitted from the main text. Sections A.1 and A.2 justify the change of measure and the weak Zakai equation, Sections A.3 and A.4 discuss Theorems 2 and 3, Section A.5 details the split update, and Sections A.6–A.8 prove Theorem 5, Proposition 7 and Corollary 6.

A.1 PROOF OF PROPOSITION 1

The density process in Proposition 1 is the Doléans exponential

$$Z_t = \exp\Bigg(-\int_0^t \eta_0(s)\,\mathrm{d}W_s^{X,\mathbb{P}} - \frac{1}{2}\int_0^t \eta_0(s)^2\,\mathrm{d}s + \int_0^t\int_{\mathbb{R}} \ln\big(1-\eta_1(s,z)\big)\,\widetilde{N}^{X,\mathbb{P}}(\mathrm{d}s,\mathrm{d}z) + \int_0^t\int_{\mathbb{R}} \big[\ln(1-\eta_1(s,z)) + \eta_1(s,z)\big]\nu^X(\mathrm{d}z)\,\mathrm{d}s\Bigg).$$

Choice of (η_0,η_1). The coefficient η_0 shifts the drift of the Brownian part of X_t, and η_1 changes the compensator of the jump measure from $\nu^X(\mathrm{d}z)\,\mathrm{d}t$ under $\mathbb{P}$ to $(1-\eta_1(t,z))\,\nu^X(\mathrm{d}z)\,\mathrm{d}t$ under $\mathbb{Q}$. The matching condition in Proposition 1 is a single equation for the pair (η_0,η_1), so the reference measure is not unique. When $\sigma>0$, the choice $\eta_1=0$ and $\eta_0=\mu/\sigma$ changes only the Brownian part. Every admissible choice yields the same filter, since the normalization $\pi_t(\varphi)=q_t(\varphi)/q_t(1)$ removes the dependence of q_t on $\mathbb{Q}$, and the Zakai equation (3) holds for any such choice.

Integrability. For $\mathbb{Q}$ to be a probability measure equivalent to $\mathbb{P}$, the Doléans exponential Z must be a true martingale on $[0,T]$ rather than only a local martingale. Let

$$V_T = \frac{1}{2}\int_0^T \eta_0(s)^2\,\mathrm{d}s + \int_0^T\int_{\mathbb{R}} \Big[\big(1-\eta_1(s,z)\big)\ln\big(1-\eta_1(s,z)\big) + \eta_1(s,z)\Big]\nu^X(\mathrm{d}z)\,\mathrm{d}s.$$

A sufficient condition is $\eta_1<1$ together with $\mathbb{E}_{\mathbb{P}}[\exp(V_T)]<\infty$ (Lépingle & Mémin, 1978), which we assume throughout.

Proof. Under the integrability condition above, the Doléans exponential Z_t is a strictly positive uniformly integrable martingale, and $1-\eta_1(t,z)>0$ ensures the jump likelihood ratio is well defined. By the Girsanov theorem for semimartingales with Brownian and jumps, the measure $\mathbb{Q}$ defined by $\mathrm{d}\mathbb{Q}/\mathrm{d}\mathbb{P}|_{\mathcal{F}_t}=Z_t$ is equivalent to $\mathbb{P}$ on each $\mathcal{F}_t$, and

$$W_t^{X,\mathbb{Q}} = W_t^{X,\mathbb{P}} + \int_0^t \eta_0(s)\,\mathrm{d}s,$$

$$\widetilde{N}^{X,\mathbb{Q}}(\mathrm{d}t,\mathrm{d}z) = \widetilde{N}^{X,\mathbb{P}}(\mathrm{d}t,\mathrm{d}z) + \eta_1(t,z)\,\nu^X(\mathrm{d}z)\mathrm{d}t.$$

Substituting these into the $\mathbb{P}$-dynamics of X_t gives a drift term

$$\mu(t,X_t,\Theta_t) - \sigma(t,X_t,\Theta_t)\eta_0(t) - \int_{\mathbb{R}} \gamma(t,z,X_{t^-},\Theta_t)\eta_1(t,z)\,\nu^X(\mathrm{d}z),$$

which vanishes by the matching condition in the proposition. Hence under $\mathbb{Q}$,

$$\mathrm{d}X_t = \sigma(t,X_t,\Theta_t)\,\mathrm{d}W_t^{X,\mathbb{Q}} + \int_{\mathbb{R}} \gamma(t,z,X_{t^-},\Theta_t)\,\widetilde{N}^{X,\mathbb{Q}}(\mathrm{d}t,\mathrm{d}z).$$

For any bounded measurable φ, Bayes' rule yields

$$\mathbb{E}_{\mathbb{P}}[\varphi(\Theta_t)\mid\mathcal{F}_t^X] = \frac{\mathbb{E}_{\mathbb{Q}}[\varphi(\Theta_t)Z_t^{-1}\mid\mathcal{F}_t^X]}{\mathbb{E}_{\mathbb{Q}}[Z_t^{-1}\mid\mathcal{F}_t^X]},$$

so the unnormalized filter is $q_t(\varphi)=\mathbb{E}_{\mathbb{Q}}[\varphi(\Theta_t)Z_t^{-1}\mid\mathcal{F}_t^X]$. □

A.2 Derivation of the weak Zakai equation (3)

For the latent process (2), the generator $\mathcal{L}_t$ acting on a smooth bounded test function φ is

$$(\mathcal{L}_t\varphi)(\theta) = a(t,\theta)\,\partial_\theta\varphi(\theta) + \frac{1}{2}b^2(t,\theta)\,\partial^2_{\theta\theta}\varphi(\theta) + \int_{\mathbb{R}}\Big[\varphi\big(\theta + k(t,\theta,z)\big) - \varphi(\theta) - k(t,\theta,z)\,\partial_\theta\varphi(\theta)\Big]\nu^{\Theta}(\mathrm{d}z).$$

Under $\mathbb{Q}$, the reciprocal density Z_t^{-1} satisfies

$$\mathrm{d}Z_t^{-1} = Z_{t^-}^{-1}\eta_0(t)\,\mathrm{d}W_t^{X,\mathbb{Q}} + \int_{\mathbb{R}} Z_{t^-}^{-1}\,\frac{\eta_1(t,z)}{1-\eta_1(t,z)}\,\widetilde{N}^{X,\mathbb{Q}}(\mathrm{d}t,\mathrm{d}z).$$

Write $\hat{\eta}_1(t,z) = \eta_1(t,z)/(1-\eta_1(t,z))$ for the effective jump coefficient of Z_t^{-1}. Since $\hat{\eta}_1$ is again a bounded predictable process, the subsequent algebra is identical. The latent process Θ_t evolves under $\mathbb{Q}$ with generator $\mathcal{L}_t$ unchanged from $\mathbb{P}$. For a smooth bounded test function φ,

$$\varphi(\Theta_t) = \varphi(\Theta_0) + \int_0^t (\mathcal{L}_s\varphi)(\Theta_s)\,\mathrm{d}s + M_t^{\varphi},$$

where M^{φ} is a $\mathbb{Q}$-martingale orthogonal to the observation noises. By Itô's product rule for semi-martingales,

$$\mathrm{d}(\varphi(\Theta_t)Z_t^{-1}) = \varphi(\Theta_{t^-})\,\mathrm{d}Z_t^{-1} + Z_{t^-}^{-1}\,\mathrm{d}\varphi(\Theta_t) + \mathrm{d}[\varphi(\Theta), Z^{-1}]_t.$$

The covariation $[\varphi(\Theta), Z^{-1}]$ vanishes because Θ and $W^{X,\mathbb{Q}}$ are independent under $\mathbb{Q}$, and joint jumps contribute only through the Poisson term already captured by the $\hat{\eta}_1$ reweighting. Applying the product rule to $\varphi(\Theta_t)Z_t^{-1}$ and taking conditional expectation with respect to $\mathcal{F}_t^X$ gives

$$\mathrm{d}q_t(\varphi) = q_t(\mathcal{L}_t\varphi)\,\mathrm{d}t + q_t(\eta_0\,\varphi)\,\mathrm{d}W_t^{X,\mathbb{Q}} + \int_{\mathbb{R}} q_{t^-}\big(\hat{\eta}_1(\cdot,z)\,\varphi\big)\,\widetilde{N}^{X,\mathbb{Q}}(\mathrm{d}t,\mathrm{d}z),$$

which is (3).

A.3 Remark on Theorem 2

Theorem 2 is a classical result. Under standard boundedness, Lipschitz, and positivity conditions on the coefficients and on the likelihood-ratio factors, the weak Zakai equation admits a unique càdlàg measure-valued solution; see Kurtz & Ocone (1988); Ceci & Colaneri (2014). We state it as a structural fact without reproducing the proof.

A.4 Remark on Theorem 3

Theorem 3 is a known approximation result for Lévy processes (Asmussen & Rosiński, 2001). In the observation model (1), the jumps of magnitude at most ε form the compensated martingale

$$J_t^{(\varepsilon)} = \int_0^t\int_{\mathbb{R}} \mathbf{1}\{|\gamma(s,z,X_{s^-},\Theta_s)| \le \varepsilon\}\,\gamma(s,z,X_{s^-},\Theta_s)\,\widetilde{N}^X(\mathrm{d}s,\mathrm{d}z),$$

which has conditional mean zero and conditional variance rate $s_t^2(\varepsilon) = \int_{\mathbb{R}} \mathbf{1}\{|\gamma| \le \varepsilon\}\,\gamma^2\,\nu^X(\mathrm{d}z)$. Over a short horizon h, Theorem 3 allows it to be replaced by a Gaussian increment with variance $s_t^2(\varepsilon)\,h$. The decoder therefore represents these jumps through its effective volatility, with σ_ϕ^2 approximating $\sigma^2 + s_t^2(\varepsilon)$, and models only jumps larger than ε explicitly through $(\lambda_\phi, \gamma_\phi, \delta_\phi)$. This approximation concerns the decoder and is independent of the encoder analysis in Theorem 5.

A.5 Details of the split update

Why the update is split. Over one observation interval $[t_k, t_{k+1}]$, the increment ΔX_k depends on the latent path through the whole interval, while the latent state itself moves according to its prior dynamics. The generator term of (3) and the innovation terms therefore act simultaneously, and their joint evolution has no closed form. Evaluating the likelihood of ΔX_k at one end of the step gives the

two one-sided schemes $q_{k+1} = \ell_k \odot \mathcal{A}_{\Delta t}(q_k)$ and $q_{k+1} = \mathcal{A}_{\Delta t}(\ell_k \odot q_k)$, where $\odot$ denotes pointwise multiplication on the latent grid. The Strang arrangement places half of the likelihood on each side,

$$q_{k+1} = \ell_k^{1/2} \odot \mathcal{A}_{\Delta t}\big(\ell_k^{1/2} \odot q_k\big),$$

so that a latent transition from θ to θ' receives the weight $\exp\big(\frac{1}{2}\log \ell_k(\theta) + \frac{1}{2}\log \ell_k(\theta')\big)$. This is the trapezoidal approximation of the log-likelihood accumulated along the latent path over the step, and it removes the leading error of the one-sided schemes in the same way as the trapezoidal rule improves on the rectangle rule (Strang, 1968).

Likelihood of one increment. Given θ, let $N_{\Delta t} \sim \mathrm{Poisson}(\lambda \Delta t)$ be the number of jumps in the step. Conditionally on $N_{\Delta t} = n$, the increment is Gaussian with mean $\mu\Delta t + n\gamma$ and variance $\sigma^2 \Delta t + n\delta^2$, so that its exact likelihood is

$$\ell_k^{\mathrm{exact}} = \sum_{n \geq 0} e^{-\lambda \Delta t} \, \frac{(\lambda \Delta t)^n}{n!} \, \mathcal{N}\big(\Delta X_k;\, \mu\Delta t + n\gamma,\, \sigma^2 \Delta t + n\delta^2\big)\,.$$

Keeping the terms $n \leq 1$ gives $\ell_k = e^{-\lambda \Delta t}\big(\mathcal{N}_0 + \lambda \Delta t \, \mathcal{N}_1\big)$ of Section 4.1. The neglected terms carry probability $\mathbb{P}(N_{\Delta t} \geq 2) \leq (\lambda \Delta t)^2/2$, which is the source of the second term in Theorem 5.

Factorization into continuous and jump parts. Dividing by the no-jump density gives

$$\ell_k = \ell_{\mathcal{B},k}\, \ell_{\mathcal{C},k}, \qquad \ell_{\mathcal{B},k} = \mathcal{N}_0, \qquad \ell_{\mathcal{C},k} = e^{-\lambda \Delta t}\left(1 + \lambda \Delta t \, \frac{\mathcal{N}_1}{\mathcal{N}_0}\right).$$

The factor $\ell_{\mathcal{B},k}$ is the likelihood of ΔX_k when no jump occurs, and $\ell_{\mathcal{C},k}$ corrects it for a possible jump. When ΔX_k is well explained by the continuous component, $\mathcal{N}_1/\mathcal{N}_0$ is small and $\ell_{\mathcal{C},k} \approx e^{-\lambda \Delta t}$. When ΔX_k lies in the tail of $\mathcal{N}_0$ but is compatible with a jump, $\mathcal{N}_1/\mathcal{N}_0$ is large and $\ell_{\mathcal{C},k}$ shifts mass toward latent values with a high jump intensity and a matching jump size.

Why $\mathcal{B}$ and $\mathcal{C}$ appear on both sides. The half-steps multiply the belief pointwise, $\mathcal{B}_h[q] = \ell_{\mathcal{B},k}^{1/2} \odot q$ and $\mathcal{C}_h[q] = \ell_{\mathcal{C},k}^{1/2} \odot q$. Pointwise multiplications commute, so $\mathcal{C}_h \circ \mathcal{B}_h$ multiplies q by $\ell_k^{1/2}$ and (4) coincides with the symmetric scheme above. Over one step, each latent value is therefore weighted by $\ell_k^{1/2}$ before and after the prior propagation, and ΔX_k enters exactly once, split between the two ends of the step. Separating $\mathcal{B}$ from $\mathcal{C}$ introduces no further splitting error. It keeps the continuous and jump contributions distinct, which allows each of them to be removed separately in the ablations of Appendix C.3.

A.6 Proof of Theorem 5

Proof. Let $h = \Delta t/2$ and define the ideal Strang update

$$\mathcal{S}_{\Delta t}^{\mathrm{Strang}} := \mathcal{C}_h^{\mathrm{exact}} \circ \mathcal{B}_h \circ \mathcal{A}_{\Delta t} \circ \mathcal{B}_h \circ \mathcal{C}_h^{\mathrm{exact}},$$

where $\mathcal{C}_h^{\mathrm{exact}}$ denotes the jump innovation built from the exact likelihood of ΔX_k, without the at-most-one-jump truncation. We decompose

$$\left\|\widetilde{\mathcal{S}}_{\Delta t}(q) - \mathcal{S}_{\Delta t}(q)\right\|_1 \leq \underbrace{\left\|\mathcal{S}_{\Delta t}^{\mathrm{Strang}}(q) - \mathcal{S}_{\Delta t}(q)\right\|_1}_{\text{(I)}} + \underbrace{\left\|\widetilde{\mathcal{S}}_{\Delta t}(q) - \mathcal{S}_{\Delta t}^{\mathrm{Strang}}(q)\right\|_1}_{\text{(II)}}.$$

(I) Splitting error. Under the regularity assumptions ensuring the L^1-stability of the filtering semigroup, Strang splitting is second-order accurate for the continuous operators; see (Strang, 1968; Bensoussan et al., 1990). Therefore,

$$\text{(I)} \leq C_1 \Delta t^2$$

for a constant $C_1 > 0$ independent of Δt.

(II) Truncation error. Let $N_{\Delta t}$ be the number of jumps larger than ε over one step. By assumption,

$$N_{\Delta t} \sim \mathrm{Poisson}(\Lambda_\varepsilon \Delta t).$$

The truncated likelihood ℓ_k in Section 4.1 coincides with the exact likelihood of ΔX_k on the event $\{N_{\Delta t} \le 1\}$ and differs only on $\{N_{\Delta t} \ge 2\}$. Writing both as expectations over the jump count,

$$\ell_k^{\text{exact}} = \mathbb{E}\big[\ell_k^{(N_{\Delta t})}\big], \qquad \ell_k = \mathbb{E}\big[\ell_k^{(\min(N_{\Delta t},1))}\big],$$

and using that each conditional update remains a normalized density, the corresponding jump innovations satisfy

$$\big\|\mathcal{C}_h^{\text{exact}}(q) - \mathcal{C}_h(q)\big\|_1 \le 2\,\mathbb{P}(N_{\Delta t} \ge 2).$$

For a Poisson random variable,

$$\mathbb{P}(N_{\Delta t} \ge 2) = 1 - e^{-\Lambda_\varepsilon \Delta t}(1 + \Lambda_\varepsilon \Delta t) \le \frac{(\Lambda_\varepsilon \Delta t)^2}{2},$$

so that

$$\big\|\mathcal{C}_h^{\text{exact}} - \mathcal{C}_h\big\|_{L^1} \le C_2(\Lambda_\varepsilon \Delta t)^2$$

for a constant $C_2 > 0$. Since the intermediate operators $\mathcal{A}_{\Delta t}$ and $\mathcal{B}_h$ are L^1-stable, propagating this defect through the Strang chain preserves the same order, so

$$\text{(II)} \le C_2(\Lambda_\varepsilon \Delta t)^2.$$

Combining (I) and (II) proves Theorem 5. □

A.7 Proof of Proposition 7

Proof. Write $\bar{q} = q/\|q\|_1$ and $\bar{\tilde{q}} = \tilde{q}/\|\tilde{q}\|_1$. Then

$$\bar{\tilde{q}} - \bar{q} = \frac{\tilde{q} - q}{\|\tilde{q}\|_1} + q\left(\frac{1}{\|\tilde{q}\|_1} - \frac{1}{\|q\|_1}\right).$$

Taking L^1 norms and using $\|q/\|\tilde{q}\|_1\|_1 = \|q\|_1/\|\tilde{q}\|_1$,

$$\|\bar{\tilde{q}} - \bar{q}\|_1 \le \frac{\|\tilde{q} - q\|_1}{\|\tilde{q}\|_1} + \frac{\|q\|_1}{\|\tilde{q}\|_1} \cdot \frac{\big|\|\tilde{q}\|_1 - \|q\|_1\big|}{\|q\|_1}.$$

Since $\big|\|\tilde{q}\|_1 - \|q\|_1\big| \le \|\tilde{q} - q\|_1$, the second term is bounded by $\|\tilde{q} - q\|_1/\|\tilde{q}\|_1$. Hence

$$\|\bar{\tilde{q}} - \bar{q}\|_1 \le \frac{2\,\|\tilde{q} - q\|_1}{\|\tilde{q}\|_1}.$$

Finally, $\|\tilde{q}\|_1 \ge \|q\|_1 - \|\tilde{q} - q\|_1$. For $\|\tilde{q} - q\|_1 \le \|q\|_1/2$ (which holds for sufficiently small Δt), we have $\|\tilde{q}\|_1 \ge \|q\|_1/2$, giving

$$\|\bar{\tilde{q}} - \bar{q}\|_1 \le \frac{2}{\|q\|_1}\|\tilde{q} - q\|_1.$$

□

A.8 Proof of Corollary 6

Proof. Let $q_k := \mathcal{S}_{\Delta t}^k(q)$ and $\tilde{q}_k := \widetilde{\mathcal{S}}_{\Delta t}^k(q)$. Define the one-step defect

$$e_{k+1} := \widetilde{\mathcal{S}}_{\Delta t}(\tilde{q}_k) - \mathcal{S}_{\Delta t}(\tilde{q}_k).$$

Then

$$\tilde{q}_{k+1} - q_{k+1} = \mathcal{S}_{\Delta t}(\tilde{q}_k) - \mathcal{S}_{\Delta t}(q_k) + e_{k+1}.$$

By Assumption 4 and Theorem 5,

$$\|\tilde{q}_{k+1} - q_{k+1}\|_1 \le (1 + L\Delta t)\,\|\tilde{q}_k - q_k\|_1 + C_1\Delta t^2 + C_2(\Lambda_\varepsilon \Delta t)^2.$$

Iterating this recursion with $\tilde{q}_0 = q_0$ yields

$$\|\tilde{q}_n - q_n\|_1 \le \sum_{j=0}^{n-1}(1 + L\Delta t)^{n-1-j}\big(C_1\Delta t^2 + C_2(\Lambda_\varepsilon \Delta t)^2\big).$$

Using $(1 + L\Delta t)^n \le e^{LT}$ and $n = T/\Delta t$, we obtain

$$\Big\|\widetilde{\mathcal{S}}_{\Delta t}^n(q) - \mathcal{S}_{\Delta t}^n(q)\Big\|_1 \le C_T(1 + \Lambda_\varepsilon^2)\Delta t$$

for some constant $C_T > 0$ depending on T but independent of Δt. This proves Corollary 6. □

B EXPERIMENTAL SETUP

B.1 DATASETS AND PREPROCESSING

Synthetic dataset. The synthetic dataset is a single trajectory of 30,000 steps with $\Delta t = 0.01$, simulated from the scalar partially observed jump-diffusion

$$\mathrm{d}\Theta_t = \kappa(\bar{\theta} - \Theta_t)\mathrm{d}t + \sigma_\theta \mathrm{d}W_t^\Theta, \qquad \mathrm{d}X_t = a_1\Theta_t\,\mathrm{d}t + \sigma_X \mathrm{d}W_t^X + c_X \mathrm{d}N_t, \qquad \lambda_t = (b_1\Theta_t)^+,$$

with $\kappa = 0.5$, $\bar{\theta} = 1.0$, $\sigma_\theta = 0.3$, $a_1 = 0.02$, $\sigma_X = 0.05$, $c_X = 0.2$ and $b_1 = 0.7$, where N_t is a counting process with intensity λ_t. Both the drift and the jump intensity of X_t depend on the latent state. Because the observation law is known, this dataset tests latent-state recovery and controlled decoder misspecification (Appendix C.4). The $\mathcal{A}$-step uses the exact Ornstein–Uhlenbeck transition kernel of Θ_t on a 401-point grid over $[-2, 2]$.

Real datasets. We use three real series with different dynamics. Figure 4 shows each series.

XAU/USD. Gold quoted against the U.S. dollar, from the Titan FX Research Hub (Titan FX, 2025). We use one-minute OHLCV records from January 2 to November 28, 2025. The Historical Data Terms of Use allow market analysis but prohibit redistribution, so we cite the source and describe the preprocessing without redistributing the records. The series has long directional stretches with local reversals and sudden price moves, which tests adaptation to regime changes.

BTC/USDT perpetual futures. The BTCUSDT USD-M perpetual futures contract on Binance, from Binance's public 15-minute klines (Binance, 2025). The sample covers January 1 to December 31, 2025, with 35,040 bars. The 15-minute increments are heavy-tailed, with excess kurtosis of about 16 on the training and validation segments, and show volatility clustering, while variance ratios close to one indicate little predictable drift, so the main difficulty is representing jumps and time-varying volatility. The chronological 70/15/15 split gives training (Jan. 1–Sep. 13, 2025), validation (Sep. 13–Nov. 7, 2025) and testing (Nov. 7–Dec. 31, 2025) segments.

MSTR perpetual futures. The MSTRUSDT USD-M perpetual futures contract on Binance, whose underlying is the stock of Strategy Inc. (formerly MicroStrategy), from Binance's public one-minute klines (Binance, 2026). The sample runs from the listing of the contract on February 9, 2026 to August 31, 2026 and contains no missing minutes. The contract trades around the clock, but 10-minute returns are about twice as volatile during U.S. exchange hours as during weekday off-hours and nearly three times as volatile as on weekends, and the price often gaps at the U.S. open. The chronological 70/15/15 split gives training (Feb. 9–Jul. 1, 2026), validation (Jul. 1–Aug. 1, 2026) and testing (Aug. 1–Aug. 31, 2026) segments.

Preprocessing. The synthetic trajectory is kept in its original scale. For each real dataset we extract a single regularly sampled positive series S_t, namely the 10-minute close for XAU/USD and MSTR, resampled from the one-minute records, and the 15-minute close for BTC/USDT. For these datasets, we apply the relative log transformation

$$X_t = \log S_t - \log S_0,$$

where S_0 is the first observation of the series. Figure 4 plots X_t, which is used for all training and evaluation.

B.2 TRAINING AND EVALUATION PROTOCOL

Forecasting setup. All experiments use context length $M = 300$, horizon $N = 100$ and stride 100. Splits are chronological, with 60/20/20 on the synthetic benchmark and 70/15/15 on the real datasets. Training windows end inside the training segment, and all metrics are computed on the test segment. Every model is trained and evaluated under five seeds $\{42, 43, 44, 45, 46\}$, which seed both training and the Monte Carlo forecast rollout, and all tables report mean $\pm$ standard deviation over the five seeds. Predictive standard deviations are used as produced by each model, with no post-hoc rescaling for any method. The time increment is $\Delta t = 0.01$ on the synthetic benchmark and $\Delta t = 10/1440$ days on XAU/USD and MSTR, and $\Delta t = 15/1440$ days on BTC/USDT.

Architecture. On the real datasets the latent state is represented on a grid over $[-1, 1]$ with $G = 25$ points on XAU/USD and $G = 50$ on BTC/USDT and MSTR, and the $\mathcal{A}$-step uses a driftless Brownian

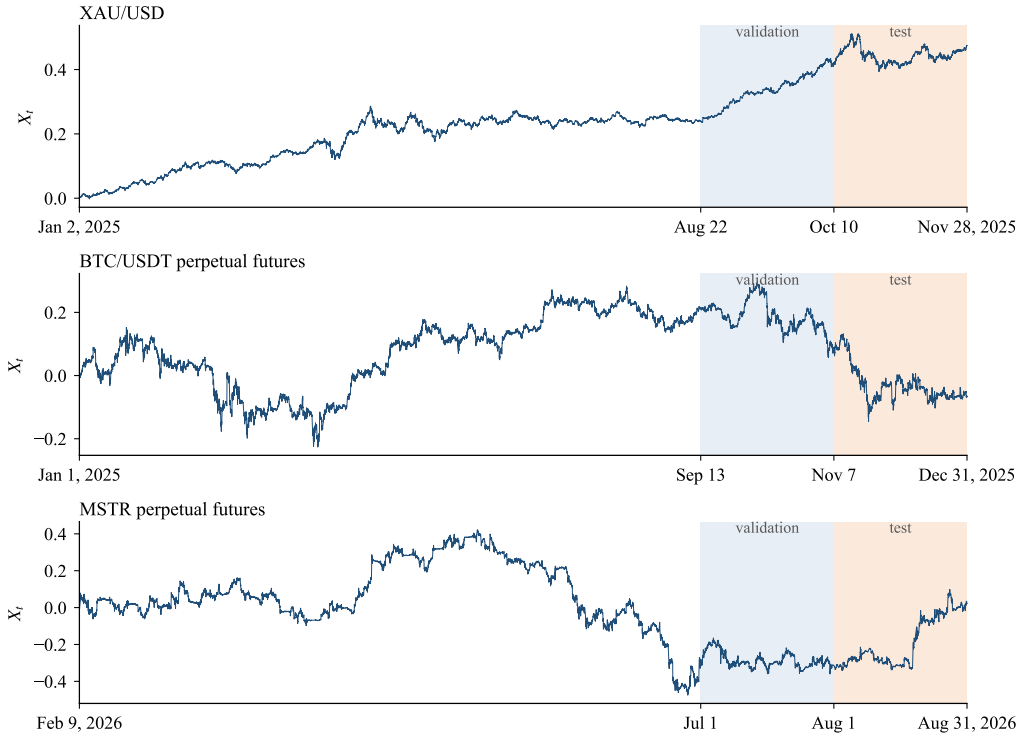


Figure 4: The three real-world series in their modeling scale, with the validation and test segments shaded.

Algorithm B.1 One filter step $\textsc{FilterStep}(q, \Delta X_k, t_k, X_{t_k})$ of the *Deep ZakaiJ* encoder.

Require: Unnormalized belief q on the grid $\{\theta_j\}_{j=1}^{G}$, increment ΔX_k, decoder ϕ, residual network ψ, reference kernel $K_{\Delta t}$.

for $j = 1, \dots, G$ **do**
  $(\mu_j, \sigma_j, \lambda_j, \gamma_j, \delta_j) \leftarrow \text{decoder}_\phi(t_k, X_{t_k}, \theta_j)$
  $\mathcal{N}_{0,j} \leftarrow \mathcal{N}(\Delta X_k;\, \mu_j \Delta t,\, \sigma_j^2 \Delta t), \quad \mathcal{N}_{1,j} \leftarrow \mathcal{N}(\Delta X_k;\, \mu_j \Delta t + \gamma_j,\, \sigma_j^2 \Delta t + \delta_j^2)$
  $\ell_{\mathcal{B},j} \leftarrow \mathcal{N}_{0,j}, \quad \ell_{\mathcal{C},j} \leftarrow e^{-\lambda_j \Delta t}\big(1 + \lambda_j \Delta t\, \mathcal{N}_{1,j}/\mathcal{N}_{0,j}\big)$
end for
$q \leftarrow q \odot \ell_{\mathcal{C}}^{1/2}, \quad q \leftarrow q \odot \ell_{\mathcal{B}}^{1/2}$ ▷ $\mathcal{C}_h$, then $\mathcal{B}_h$
$u \leftarrow q, \quad p(\theta_j) \leftarrow \sum_{i=1}^{G} K_{\Delta t}(\theta_j \mid \theta_i)\, u(\theta_i)\, \Delta\theta$ ▷ reference propagation
$q \leftarrow \big[p + R_\psi^{\mathcal{A}}(u; \Delta t)\big]^+$ ▷ $\mathcal{A}_{\Delta t}$
$q \leftarrow q \odot \ell_{\mathcal{B}}^{1/2}, \quad q \leftarrow q \odot \ell_{\mathcal{C}}^{1/2}$ ▷ $\mathcal{B}_h$, then $\mathcal{C}_h$
$q \leftarrow q / \sum_j q(\theta_j) \Delta\theta$ ▷ rescaling
Output: $q, \quad \pi^{\text{prior}} \leftarrow p / \sum_j p(\theta_j) \Delta\theta, \quad \ell \leftarrow \ell_{\mathcal{B}} \odot \ell_{\mathcal{C}}$

reference kernel with $\sigma_\theta = 0.3$, whose mismatch with the true latent dynamics is absorbed by the residual $R_\psi^{\mathcal{A}}$. The decoder is fully neural, with hidden dimension $d_{\text{model}} = 32$, two transformer encoder layers with two attention heads, and a local-statistics window of 20 increments. The network that initializes the latent belief has layers $[8, 64, G]$, each decoder branch (drift, diffusion, jump) has a 32-unit hidden layer, and the residual network in the $\mathcal{A}$-step has hidden width 128. On the synthetic benchmark the main model uses a linear observation decoder with the grid of Appendix B.1. The polynomial decoder in Appendix C.4 uses $d_{\text{model}} = 32$ with separate drift, volatility and jump transformers of depths $(2, 1, 1)$ and two attention heads; the neural decoder has the same two-layer, two-head architecture as on the real datasets.

Optimization. The model is trained end-to-end in PyTorch with AdamW and batch size 32. The residual network uses learning rate 10^{-3} and weight decay 10^{-4}; the observation decoder uses

Algorithm B.2 Training of *Deep ZakaiJ*.

Require: Training windows $X_{s:s+M+N}$, decoder ϕ, encoder ψ (residual network and initial-belief network g_ψ), context length M, horizon N.

for each minibatch of windows **do**
  $q_0 \leftarrow \mathrm{softmax}\big(g_\psi(X_{s:s+M})\big)/\Delta\theta$ ▷ initial belief from context statistics
  $\mathcal{L} \leftarrow 0$
  for $k = 0, \dots, M-1$ **do** ▷ filtering over the context
    $\pi_k \leftarrow q_k / \sum_j q_k(\theta_j)\Delta\theta$
    $(q_{k+1},\, \pi^{\mathrm{prior}}_{k+1},\, \ell_k) \leftarrow \textsc{FilterStep}(q_k, \Delta X_k, t_k, X_{t_k})$
    $\mathcal{L} \leftarrow \mathcal{L} + \sum_j \pi_k(\theta_j) \log \ell_k(\theta_j)\, \Delta\theta - \mathrm{KL}\big(\pi_{k+1} \,\|\, \pi^{\mathrm{prior}}_{k+1}\big)$
  end for
  for $k = M, \dots, M+N-1$ **do** ▷ forecasting over the target segment
    $\pi_k \leftarrow q_k / \sum_j q_k(\theta_j)\Delta\theta$
    $\mathcal{L} \leftarrow \mathcal{L} + \sum_j \pi_k(\theta_j) \log p_\phi(\Delta X_k \mid t_k, X_{t_k}, \theta_j)\, \Delta\theta$
    $q_{k+1} \leftarrow \mathcal{A}_{\Delta t}[q_k]$ ▷ prior propagation only
  end for
  Update (ϕ, ψ) by a gradient step on $-\mathcal{L}$.
end for

Algorithm B.3 Inference of *Deep ZakaiJ*.

Require: Trained (ϕ, ψ), context $X_{s:s+M}$, horizon N, number of trajectories S.

$q_0 \leftarrow \mathrm{softmax}\big(g_\psi(X_{s:s+M})\big)/\Delta\theta$
for $k = 0, \dots, M-1$ **do**
  $q_{k+1} \leftarrow \textsc{FilterStep}(q_k, \Delta X_k, t_k, X_{t_k})$
end for
for $m = 1, \dots, S$ **do**
  $q \leftarrow q_M, \quad x \leftarrow X_{t_M}$
  for $n = 0, \dots, N-1$ **do**
    Sample $\theta \sim q / \sum_j q(\theta_j)\Delta\theta$
    $(\mu, \sigma, \lambda, \gamma, \delta) \leftarrow \mathrm{decoder}_\phi(t_{M+n}, x, \theta)$
    Sample $J \sim \mathrm{Poisson}(\lambda\Delta t)$, $Y_i \sim \mathcal{N}(\gamma, \delta^2)$ for $i \le J$, and $\xi \sim \mathcal{N}(0,1)$
    $x \leftarrow x + \mu\Delta t + \sigma\sqrt{\Delta t}\,\xi + \sum_{i=1}^{J} Y_i, \quad X^{(m)}_{t_{M+n+1}} \leftarrow x$
    $q \leftarrow \mathcal{A}_{\Delta t}[q]$ ▷ prior propagation only
  end for
end for
Output: Ensemble $\{X^{(m)}_{t_{M+1}:t_{M+N}}\}_{m=1}^{S}$.

learning rate 5×10^{-4} and weight decay 10^{-3}. Training runs for 100 epochs on MSTR and 50 epochs on the other datasets, with a 3-epoch warm-up followed by cosine decay. Gradient norms are clipped at 10.0 for the residual network and 1.0 for the decoder.

Forecast evaluation. *Deep ZakaiJ* generates 100 Monte Carlo trajectories per window by rolling out the learned one-step jump-diffusion law (Algorithm B.3). MAE and RMSE are computed from the point forecast of the ensemble, and CRPS, average log-likelihood and IS90 from the predictive distribution induced by the 100 trajectories. All baselines use the same splits, windows and metrics. The trained baselines follow their reference implementations with input and output lengths set to our protocol. The foundation models are used zero-shot with the pretrained checkpoints Chronos-T5-small, TimesFM-1.0-200M, Moirai-MoE-R-small and Lag-Llama.

B.3 Computational cost

All five-seed experiments were run on a single NVIDIA GeForce RTX 5090 GPU. Table 3 reports training and inference wall-clock time for every model under the same rolling-evaluation protocol.

Table 3: Training and inference wall-clock time in seconds on a single RTX 5090.

Model	Synthetic		XAU/USD		MSTR		BTC/USDT	
	Train	Infer	Train	Infer	Train	Infer	Train	Infer
DLinear	0	0	1	0	1	0	1	0
LSTM	4	13	4	10	5	9	5	11
DeepAR	3	204	4	163	3	151	4	184
PatchTST	4	0	7	0	7	0	7	0
LatentSDE	32	225	41	162	42	148	42	177
LatentODE	150	154	184	106	188	99	187	117
NCDSSM	157	15	195	11	204	10	205	12
NJ-ODE	138	9	172	7	179	6	180	7
NeuralMJD	3	1	5	1	5	1	5	1
Chronos	3	69	4	55	4	49	4	58
GRU-ODE	174	19	217	14	227	12	229	15
S4	7	21	11	16	10	15	10	18
TimesFM	4	2	2	2	2	2	3	2
Moirai-MoE	4	34	2	25	2	23	3	28
Lag-Llama	1	141	1	116	1	103	1	124
iTransformer	5	10	6	8	6	7	6	9
PF-RNN	184	26	233	18	236	17	247	21
Deep ZakaiJ	729	126	1183	194	2407	182	1458	221

Latent dimension. The $\mathcal{A}$-step propagates the belief on a grid with G points per axis, so memory and propagation cost grow with the latent dimension d. Table 4 trains and evaluates the model with $d \in \{1, 2, 3\}$ at $G = 25$ points per axis, 15 epochs and 100 forecast samples.

Table 4: Latent-dimension scaling at $G = 25$ points per axis.

d	Grid cells	Parameters	Train (s)	Infer (ms/window)
1	25	27,775	581.5	102.6
2	625	206,623	585.5	112.6
3	15,625	4,676,671	606.6	323.6

C Additional results

C.1 Additional baselines

Table 5 reports the baselines not shown in the main tables.

Table 5: Additional baselines, mean ± standard deviation over five seeds.

Model	MAE	RMSE	CRPS	LogLik	IS90
Synthetic					
GRU-ODE	0.1174±.0107	0.1651±.0118	0.0870±.0075	0.02±.17	0.879±.058
S4	0.1190±.0014	0.1667±.0018	–	–	–
TimesFM	0.1235±.0000	0.1915±.0000	0.0981±.0000	−2.13±.00	1.165±.000
Moirai-MoE	0.1211±.0006	0.1916±.0010	0.1467±.0082	−0.26±.04	1.737±.110
Lag-Llama	0.3078±.0014	0.4115±.0019	0.4462±.0006	−1.24±.00	5.681±.009
iTransformer	0.1045±.0031	0.1482±.0030	–	–	–
PF-RNN	0.1243±.0073	0.1740±.0071	0.1067±.0058	0.27±.07	1.144±.067
Deep ZakaiJ	**0.0850**±.0009	**0.1367**±.0017	**0.0671**±.0007	**0.73**±.04	**0.620**±.025
XAU/USD					
GRU-ODE	0.0080±.0003	0.0115±.0005	0.0058±.0002	3.18±.03	0.0487±.0012
S4	0.0079±.0001	0.0112±.0002	–	–	–
TimesFM	0.0087±.0000	0.0124±.0000	0.0065±.0000	2.49±.00	0.0637±.0000
Moirai-MoE	0.0078±.0001	0.0111±.0001	0.0079±.0002	2.94±.01	0.0852±.0028
Lag-Llama	0.0132±.0001	0.0173±.0001	0.0153±.0000	1.95±.00	0.1879±.0002
iTransformer	0.0110±.0002	0.0145±.0002	–	–	–
PF-RNN	0.0082±.0005	0.0117±.0007	0.0061±.0003	3.16±.05	0.0511±.0024
Deep ZakaiJ	**0.0075**±.0001	**0.0107**±.0001	**0.0054**±.0001	**3.32**±.01	**0.0433**±.0004
BTC/USDT					
GRU-ODE	0.0108±.0002	0.0166±.0002	0.0088±.0001	2.43±.04	0.0886±.0019
S4	0.0122±.0001	0.0174±.0001	–	–	–
TimesFM	0.0124±.0000	0.0181±.0000	0.0093±.0000	1.94±.00	0.0918±.0000
Moirai-MoE	0.0110±.0000	0.0167±.0000	0.0168±.0095	2.51±.02	0.2017±.1335
Lag-Llama	0.0225±.0001	0.0303±.0002	0.0268±.0001	1.43±.00	0.3315±.0005
iTransformer	0.0150±.0002	0.0218±.0005	–	–	–
PF-RNN	0.0112±.0002	0.0167±.0004	0.0093±.0004	2.52±.09	0.0937±.0058
Deep ZakaiJ	**0.0105**±.0001	**0.0158**±.0001	**0.0081**±.0001	**2.87**±.01	**0.0737**±.0014

C.2 MSTR PERPETUAL FUTURES

Table 6 reports the MSTR perpetual-futures series of Appendix B.1, evaluated under the same protocol and against the same baselines. *Deep ZakaiJ* attains the best CRPS, LogLik and IS90, at a small cost in point accuracy, where several baselines, most notably Chronos, achieve lower MAE and RMSE. For a stochastic series with frequent jumps, a point forecast summarizes a wide predictive distribution by a single path and says little about the risk of large moves, so the probabilistic metrics are the more informative comparison.

Table 6: Results on the MSTR perpetual-futures series, mean ± standard deviation over five seeds.

Model	MAE	RMSE	CRPS	LogLik	IS90
DLinear	0.0209±.0002	0.0315±.0003	–	–	–
LSTM	0.0208±.0014	0.0325±.0027	–	–	–
DeepAR	0.0965±.0964	0.1216±.1146	0.0761±.0759	0.16±1.52	0.8016±.9313
PatchTST	0.0235±.0011	0.0352±.0019	–	–	–
LatentSDE	0.0175±.0007	0.0269±.0015	0.0141±.0004	1.63±.09	0.1500±.0035
LatentODE	0.0259±.0010	0.0397±.0008	0.0194±.0006	0.87±.07	0.1913±.0081
NCDSSM	0.0187±.0006	0.0301±.0009	0.0148±.0003	0.07±.11	0.1643±.0023
NJ-ODE	0.0178±.0002	0.0255±.0006	–	–	–
NeuralMJD	0.0165±.0005	0.0246±.0005	0.0137±.0006	2.26±.03	0.1343±.0075
Chronos	**0.0160**±.0002	**0.0239**±.0004	0.0171±.0006	−5.4e4±1.2e5	0.2217±.0102
GRU-ODE	0.0169±.0004	0.0256±.0010	0.0135±.0003	2.06±.10	0.1345±.0017
S4	0.0169±.0002	0.0243±.0002	–	–	–
TimesFM	0.0203±.0000	0.0316±.0000	0.0157±.0000	−1.51±.00	0.1692±.0000
Moirai-MoE	0.0170±.0001	0.0267±.0002	0.0154±.0008	2.29±.02	0.1554±.0124
Lag-Llama	0.0274±.0001	0.0427±.0002	0.0219±.0001	−81.18±8.69	0.2229±.0028
iTransformer	0.0193±.0008	0.0270±.0010	–	–	–
PF-RNN	0.0179±.0007	0.0283±.0018	0.0167±.0005	2.02±.06	0.1809±.0075
Deep ZakaiJ	0.0166±.0002	0.0256±.0005	**0.0128**±.0002	**2.29**±.06	**0.1192**±.0030

C.3 ABLATIONS

We evaluate the contribution of each architectural component on the synthetic benchmark, where the latent state drives both the drift and the jump intensity of the observations, so that each component

Table 7: Ablations on the synthetic benchmark, mean ± standard deviation over five seeds.

Variant	MAE	RMSE	CRPS	LogLik	IS90
Decoder-only	0.0855 ±.0010	0.1421±.0008	0.0687±.0005	0.71±.01	0.645±.006
GRU belief	0.0857±.0008	0.1379±.0014	0.0675±.0008	0.68±.08	0.629±.008
Without $\mathcal{A}$	0.0861±.0011	0.1377±.0018	0.0676±.0011	0.70±.09	0.624±.016
Without $\mathcal{B}$	0.0863±.0010	0.1379±.0021	0.0676±.0010	0.69±.10	0.623±.018
Without $\mathcal{C}$	0.0862±.0012	0.1379±.0025	0.0676±.0013	0.68±.10	0.624±.019
Without $R_\psi^{\mathcal{A}}$	0.0867±.0010	0.1384±.0016	0.0679±.0009	0.68±.11	0.628±.016
Deep ZakaiJ (full)	**0.0850**±.0009	**0.1367**±.0017	**0.0671**±.0007	**0.73**±.04	**0.620**±.025

can be assessed against a known data-generating process. As shown in Table 7, the full model is best on every metric, and the ablations are organized into two groups.

Filtering necessity. Removing the Zakai encoder entirely (Decoder-only) raises RMSE from 0.1367 to 0.1421, CRPS from 0.0671 to 0.0687 and IS90 from 0.620 to 0.645. Replacing the structured Zakai filter with a GRU belief of similar parameter count also degrades every metric, with CRPS 0.0675 and LogLik 0.68 against 0.0671 and 0.73 for the full model. Together, these confirm that latent filtering is the primary driver of forecast quality, and that the structured splitting update provides a stronger inductive bias than a generic recurrent alternative.

Splitting components. Removing any single substep ($\mathcal{A}$, $\mathcal{B}$, or $\mathcal{C}$) or the learned residual $R_\psi^{\mathcal{A}}$ degrades every metric as well, with CRPS between 0.0676 and 0.0679 and LogLik between 0.68 and 0.70, against 0.0671 and 0.73 for the full model. The residual correction has the largest effect among these components, raising MAE from 0.0850 to 0.0867 and RMSE from 0.1367 to 0.1384.

C.4 Decoder misspecification and grid resolution

Table 8 varies the observation decoder and the grid size on the synthetic benchmark, where the linear decoder is correctly specified. All rows come from one set of runs, with the linear decoder at $G = 401$ as the reference.

Table 8: Decoder family and latent grid size on the synthetic benchmark, mean ± standard deviation over five seeds.

Decoder	G	MAE	RMSE	CRPS	LogLik	IS90
Linear (reference)	401	0.0861±.0012	0.1377±.0024	0.0675±.0014	0.69±.10	0.624±.020
Polynomial	401	0.0858±.0021	0.1374±.0018	0.0689±.0009	0.53±.07	0.708±.020
Neural	401	0.0892±.0029	0.1378±.0018	0.0714±.0017	0.30±.18	0.733±.019
Linear	201	0.0868±.0006	0.1389±.0011	0.0680±.0004	0.68±.04	0.633±.011
Linear	101	0.0857±.0007	0.1377±.0015	0.0675±.0010	0.70±.03	0.629±.015
Linear	51	0.0862±.0008	0.1367±.0015	0.0674±.0003	0.71±.05	0.615±.012

Decoder misspecification. Replacing the linear decoder with a polynomial or a fully neural one leaves point accuracy nearly unchanged (MAE 0.0858 and 0.0892 against 0.0861) but degrades the predictive distribution. LogLik falls from 0.69 to 0.53 and 0.30, and IS90 rises from 0.624 to 0.708 and 0.733. The structured decoder therefore matters most for the predictive distribution.

Grid resolution. Reducing the grid from 401 to 51 points changes no metric by more than one standard deviation (CRPS between 0.0674 and 0.0680). For a scalar latent state a coarse grid is sufficient, consistent with the grids of 25 and 50 points used on the real datasets.

C.5 Sensitivity to architecture and latent domain

Table 9 varies the encoder depth, the coefficient-network width and the bounds of the latent domain on the synthetic benchmark. Using one layer fewer or more than the default, half or twice the default width, or the latent domains $[-1.5, 1.5]$ and $[-3, 3]$ in place of $[-2, 2]$ keeps CRPS between 0.0671 and 0.0678, against 0.0671 for the default configuration in Table 1. Narrowing the latent domain has the largest effect, lowering LogLik from 0.73 to 0.64. Every configuration still outperforms every

baseline of Table 1 in MAE, CRPS, LogLik and IS90, so the gains of *Deep ZakaiJ* do not rely on a finely tuned architecture.

Table 9: Sensitivity to architecture and latent domain on the synthetic benchmark, mean ± standard deviation over five seeds.

Setting	MAE	RMSE	CRPS	LogLik	IS90
Depth 2	0.0855±.0014	0.1374±.0030	0.0673±.0014	0.70±.05	0.627±.026
Depth 4	0.0855±.0011	0.1371±.0018	0.0671±.0008	0.72±.04	0.620±.020
Width 32	0.0863±.0011	0.1385±.0018	0.0677±.0010	0.67±.07	0.628±.009
Width 128	0.0854±.0018	0.1370±.0026	0.0673±.0012	0.70±.07	0.627±.023
Domain $[-1.5, 1.5]$	0.0864±.0010	0.1385±.0019	0.0678±.0011	0.64±.08	0.632±.016
Domain $[-3, 3]$	0.0864±.0013	0.1379±.0006	0.0678±.0005	0.69±.05	0.627±.010

C.6 Belief response at change points

We detect change points independently in the filtered posterior mean and in a model-free rolling bipower log-volatility series, using the same penalized change-in-mean segmentation, and match the two sets one-to-one within 25 steps. Table 10 reports the resulting F1 score, the harmonic mean of the matching precision and recall. The shift-null F1 is obtained by circularly shifting the belief change points and repeating the matching, which preserves their number and spacing while removing temporal alignment.

At each belief change point we also compute the signed before–after changes $\Delta\theta$ and $\Delta \log \mathrm{volatility}$. Let $T_{\mathrm{volatility}}$ be the mean $|\Delta \log \mathrm{volatility}|$ across these change points. The volatility-shift statistic is

$$z_{\mathrm{volatility}} = \frac{T_{\mathrm{volatility}} - \mathbb{E}_{\mathrm{null}}[T_{\mathrm{volatility}}]}{\mathrm{sd}_{\mathrm{null}}(T_{\mathrm{volatility}})},$$

where the null uses change points detected from phase-randomized belief paths, which preserve the spectrum but destroy alignment with the observations. Finally, $|\rho_S|$ is the absolute Spearman correlation between the signed $\Delta\theta$ and $\Delta \log \mathrm{volatility}$ across change points; we report the absolute value because the orientation of the learned regime coordinate is not identified, and the corresponding pairing permutation test has $p < 0.05$ in all five seeds on every dataset.

Table 10: Agreement between belief and volatility change points with a 25-step matching tolerance, mean ± standard deviation over five seeds.

Dataset	Matched F1	Shift-null F1	Volatility-shift z	$\lvert\rho_S(\Delta\theta, \Delta \log \mathrm{volatility})\rvert$
Synthetic	0.610±.028	0.211±.009	18.8±.7	0.812±.004
XAU/USD	0.776±.018	0.417±.005	6.9±.5	0.864±.009
BTC/USDT	0.428±.067	0.226±.008	11.7±.2	0.904±.028
MSTR	0.425±.018	0.206±.006	8.7±.5	0.892±.016

C.7 Empirical checks of the theoretical assumptions

Jump intensity. The truncation term of Theorem 5 depends on the rate of large jumps per step, $\Lambda_\varepsilon \Delta t$. Table 11 reports its empirical counterpart, the fraction of test-segment increments with $|\Delta X_k| > c\,\hat{\sigma}$ for $c \in \{3, 5\}$, where $\hat{\sigma}$ is the bipower-variation estimate of the per-step volatility on the training segment, which is robust to jumps. All rates are below 0.06, so the truncation term $(\Lambda_\varepsilon \Delta t)^2$ is of order 10^{-3} or smaller. Table 12 retrains the model on two additional synthetic datasets generated with the jump coefficient b_1 scaled down to 0.2 (sparse) and up to 2.8 (dense) from the benchmark value 0.7.

Table 11: Empirical large-increment rate $\hat{\Lambda}\Delta t$.

Dataset	$3\hat{\sigma}$	$5\hat{\sigma}$
Synthetic	0.0063	0.0063
XAU/USD	0.0571	0.0123
BTC/USDT	0.0188	0.0036
MSTR	0.0189	0.0041

Table 12: Jump-intensity stress test on the synthetic benchmark.

Regime	MAE	RMSE	CRPS	LogLik	IS90
Sparse	0.0464±.0004	0.0753±.0005	0.0360±.0004	1.18±.17	0.380±.013
Dense	0.1805±.0021	0.2399±.0029	0.1280±.0016	0.15±.02	0.924±.019

Stability. Assumption 4 is a property of the exact filtering semigroup and cannot be certified from finitely many trajectories. As an empirical diagnostic of the implemented update, we perturb the normalized belief within the probability simplex at randomly selected filtering states and record the one-step expansion ratio

$$\hat{L} \;=\; \frac{\left\|\widetilde{\mathcal{S}}_{\Delta t}(q_1) - \widetilde{\mathcal{S}}_{\Delta t}(q_2)\right\|_1}{\|q_1 - q_2\|_1}.$$

Table 13 reports its median, 95th percentile and maximum.

Table 13: Empirical one-step expansion ratio $\hat{L}$ of the implemented update.

Dataset	Perturbation pairs	Median	95th percentile	Maximum
Synthetic	960	0.267	0.368	0.510
XAU/USD	784	0.490	0.726	0.998
BTC/USDT	848	0.478	0.732	0.957
MSTR	704	0.453	0.722	2.029

D LIMITATIONS

The current formulation assumes univariate observations. Scaling to multivariate systems will likely require structured approximations such as tensor decompositions or amortized neural surrogates for the prior propagation step. The encoder relies on a fixed uniform latent grid, whose cost in the $\mathcal{A}$-step and rigid support may limit scalability and expressivity, replacing it with adaptive or particle-based representations could better capture multi-modal regimes. Finally, the filtering framework currently conditions only on past observations of the target series. Incorporating exogenous covariates into the belief update could broaden applicability to domains where external drivers are only partially observed, such as macroeconomic indicators in finance or weather-driven demand in electricity markets.

E IMPACT STATEMENT

This paper presents a filtering-based framework for forecasting time series with latent regimes and jumps. Potential applications include financial risk management and energy market operations (e.g., for electricity price spikes) through improved uncertainty quantification. However, overreliance on model-generated predictive intervals in high-stakes settings carries risks, particularly under distribution shifts or violated assumptions. We advise practitioners to treat these forecasts as complementary tools and to validate their calibration on domain-specific data prior to deployment.