Stochastic Dynamical Models for Financial Market Prediction and Analysis

Felix Pei
Neuroscience Institute
Carnegie Mellon University
Pittsburgh, PA 15213
fcpei@andrew.cmu.edu

Abstract

Stochastic differential equations have traditionally been used to model stock prices. However, conventional deep learning methods for timeseries forecasting typically do not explicitly model stochastic dynamics. Here, we explore the application of neural network-based stochastic dynamical models to stock market data for both forecasting and analysis of market regimes. We find that our stochastic dynamical models are competitive in financial timeseries forecasting, especially for long-horizon predictions. Additionally, the models accurately capture fluctuations in market volatility through their inferred diffusion dynamics. These stochastic dynamical models can thus serve as useful tools for forecasting stock prices and interrogating the dynamics underlying market fluctuations.

1 Introduction

Financial markets exhibit complex and noisy dynamics driven by a combination of structural market forces and stochastic variability. Traditional financial modeling often uses stochastic differential equations (SDEs) to capture these dynamics, particularly for modeling volatility and option pricing [1, 2]. Such models can provide valuable insights into market dynamics and inform trading decisions. However, inference of these models from data is challenging, often requiring expensive approximations or complex fitting procedures [3].

In contrast, many machine learning applications to finance focus on forecasting stock prices or other financial data [4]. As a result, work in this direction has relied on developments in machine learning for general timeseries forecasting. Models like DeepAR [5], PatchTST [6], Chronos [7], and Moirai [8] have been highly successful on financial timeseries forecasting benchmarks [9], significantly outperforming more classical methods like ARIMA. However, these models are typically optimized for short-term predictive accuracy rather than modeling the underlying data-generating process. As such, they often lack interpretability and do not explicitly model market dynamics.

In other domains, generative models of dynamical phenomena have led to new insights into the nature of complex systems, for example in fluid physics [10] and neuroscience [11, 12]. Thus, such approaches may also better our understanding of the complexities of market dynamics. However, because financial markets are inherently volatile and uncertain, using stochastic models may be essential for accurately capturing financial dynamics. In particular, we hypothesize that explicitly decoupling structured market fluctuations, stochastic variability in dynamics, and observation noise in the models will improve long-horizon prediction and facilitate analysis of intrinsic market dynamics.

In this work, we explore the application of stochastic dynamical models to financial data for both prediction and analysis. Specifically, we build on work on probabilistic state-space models [13, 14]

CMU 10-716 Spring 2025 Course Project

and neural SDEs [15, 16, 17] to fit stochastic dynamical models to the FNSPID dataset [18]. We evaluate the utility of these models on stock price forecasting as well as market volatility analysis.

2 Background

Our methods are closely related to discrete-time probabilistic state-space models and neural stochastic differential equations, both of which allow for the explicit modeling of stochastic dynamical systems. We briefly introduce these model classes here.

2.1 Probabilistic state-space models

Though the term state-space model (SSM) is now more commonly used for specific deep learning architectures like S4 and Mamba, the term can broadly refer to any model that has a hidden state that evolves over time in a Markovian manner—i.e., its state at one timepoint depends only on the state at the previous timepoint. Probabilistic SSMs, where the dynamics of the hidden state are stochastic, are well-established tools in statistics and control systems [19]. However, traditional probabilistic SSMs often assume linear dynamics and Gaussian noise, making them ill-suited for many real-world applications. Extensions of probabilistic SSMs to nonlinear dynamics exist (e.g., [20]), but they are often difficult to train and fail to match the performance of deterministic deep learning approaches on conventional tasks.

Fortunately, recent developments have introduced more scalable and expressive probabilistic SSMs. For instance, the Probabilistic Recurrent State-Space Model (PR-SSM) proposed in [13] combines Gaussian processes with doubly stochastic variational inference to enable robust and efficient training on large datasets. Similarly, [14] introduced the Recurrent State Space Model (RSSM) in their PlaNet framework, which integrates deterministic and stochastic components in the transition model and employs a tractable multi-step variational inference objective. These modern approaches leverage gradient-based optimization and variational inference to train nonlinear probabilistic models end-to-end.

2.2 Neural ODEs and neural SDEs

SSMs are typically discrete-time models that operate on a particular fixed temporal resolution, limiting their ability to handle irregularly-sampled data and generalize to different temporal resolutions. Neural ODEs overcome this issue by directly learning the flow governing observed dynamics and using robust ODE solvers to compute future states [15]. The critical innovation to enable scalable training of neural ODEs is the use of the adjoint method to compute gradients by solving an augmented ODE backwards in time.

Later work has extended deterministic neural ODEs to stochastic differential equations (SDEs) [16, 17, 21]. In particular, [17] derive a stochastic variant of the adjoint method to fit models of the form

$$z_T = z_0 + \int_0^T h_\theta(z_t, t)dt + \int_0^T g_\psi(z_t, t) \circ dW_t$$

where h_θ is the drift function, g_ψ is the diffusion function, and W_t is a Wiener process. Importantly, these models thus explicitly decouple observed dynamics into a deterministic drift term and a stochastic diffusion term, potentially facilitating independent analyses of structured dynamics and stochastic variability.

3 Methods

We build on probabilistic state-space models and neural SDEs to formulate a stochastic dynamical model to apply to financial timeseries data. Specifically, we use the framework of latent neural SDEs introduced by [17], where prior and posterior SDEs can be trained similar to a VAE. We discretize this framework to yield a probabilistic SSM which we term a stochastic residual dynamics model (SRDM).

3.1 Latent neural SDEs

The authors of [17] extend standard neural SDEs to latent neural SDEs, where dynamics unfold in a latent space, not the data space. To fit these latent SDEs, the authors define a prior process and posterior process:

$$
\begin{aligned}
dz_t &= h_\theta(z,t)dt + g_\psi(z,t)dW_t, && \text{(prior dyn.)} \\
dz_t &= h_\phi(z,t)dt + g_\psi(z,t)dW_t, && \text{(posterior dyn.)}
\end{aligned}
$$

where the two processes share the diffusion function g_ψ to ensure that the KL divergence is finite.

The authors derive an ELBO using these prior and posterior processes:

$$
\log p(x_{0:T}|\theta) \geq \mathbb{E}_{z_{0:T}} \left[\sum_{t=0}^{T} \log p(x_t|z_t) - \int_0^T \frac{1}{2} \|u(z_t,t)\|_2^2 dt \right]
$$

where u satisfies

$$
u(z,t) = \frac{h_\phi(z,t) - h_\theta(z,t)}{g_\psi(z,t)}.
$$

Noting that the analytical KL divergence between two Gaussians with equal variance σ^2 is

$$
D_{KL}(p \,||\, q) = \frac{(\mu_p - \mu_q)^2}{2\sigma^2},
$$

one can see that the integral $\int \frac{1}{2}\|u(z_t,t)\|^2 dt$ essentially computes the KL divergence between the prior and posterior trajectories. The exact derivation of this ELBO relies on stochastic calculus and can be found in the appendix of their paper.

3.2 Discrete time stochastic residual dynamics models (SRDM)

For typical financial datasets, which only record discrete statistics like stock opening and closing prices, the continuous-time formulation of latent neural SDEs is not necessary. Instead, we can formulate a similar discrete-time model, which we will call a stochastic residual dynamics model (SRDM):

$$
\begin{aligned}
p(x_t|z_t) &= \mathcal{N}(f_\psi(z_t), \sigma_{obs}^2) && \text{(observation)} \\
p(z_0) &= \mathcal{N}(\mu_0, \sigma_0^2) && \text{(prior init. cond.)} \\
p(z_t|z_{t-1}) &= \mathcal{N}(z_{t-1} + h_\theta(z_{t-1}), g_\psi(z_{t-1})) && \text{(prior dyn.)} \\
q(z_0|x) &= \mathcal{N}(\mu_{\phi(x)}, \sigma_{\phi(x)}^2) && \text{(posterior init. cond.)} \\
q(z_t|z_{t-1},x) &= \mathcal{N}(z_{t-1} + h_{\phi(x)}(z_{t-1}), g_\psi(z_{t-1})) && \text{(posterior dyn.)}
\end{aligned}
$$

In practice, the posterior drift function $h_{\phi(x)}$ can be parametrized as a feedforward neural network that takes as input the past latent state z_{t-1} and a context vector from an encoder. This context vector can either be time-varying (which we call "full" context) or constant across the whole sequence (which we call "constant" context). It is also possible to fix the posterior drift function to the prior drift function and only allow the encoder to specify the distribution of the initial condition (which we call "IC-only" context), resulting in a sequential VAE similar to LFADS [22]. Further, for forecasting, we can either continue the trajectory using the prior drift function or posterior drift function (where we repeat the last context vector for future timesteps), though for "IC-only" context models, the prior and posterior drift are identical. We include a graphical depiction of these architectural variants in Appendix A.1.

We can derive the standard ELBO for this model:

$$
\begin{aligned}
\log p(x) &= \log \int_z p(x|z)p(z)dz \\
&= \log \int_z p(x|z)\frac{p(z)}{q(z|x)}q(z|x)dz \\
&= \log \mathbb{E}_{z\sim q(\cdot|x)}\left[p(x|z)\frac{p(z)}{q(z|x)}\right] \\
&\geq \mathbb{E}_{z\sim q(\cdot|x)}\left[\log p(x|z) + \log\frac{p(z)}{q(z|x)}\right] \\
&= \mathbb{E}_{z\sim q(\cdot|x)}\left[\sum_{t=0}^{T}\log p(x_t|z_t) + \log\frac{p(z_0)}{q(z_0|x)} + \sum_{t=1}^{T}\log\frac{p(z_t|z_{t-1})}{q(z_t|z_{t-1},x)}\right]
\end{aligned}
$$

where we denote $x = \{x_0, \ldots, x_T\}$ and $z = \{z_0, \ldots, z_T\}$. Though the expectation over the trajectories from the posterior is intractable, we can simply rely on a Monte Carlo estimator for this expression, which may be sufficiently reliable for shorter sequences. The log-probability-ratio terms of the KL divergence can either be calculated exactly for a given sample or substituted with the analytical KL of two Gaussians, given z_{t-1}, as both are unbiased estimators of the overall expectation.

We note that our SRDM model is similar to probabilistic state space models discussed in Section 2.1. However, it differs from previous approaches in the use of a residual connection in the transition function and in separately parametrizing the prior drift, posterior drift, and diffusion terms.

4 Results

4.1 Single-stock n-step forecasting

Following the experimental setup from [18], we trained and evaluated models on single-stock n-step forecasting. The models were provided as input all available stock and sentiment features for a single stock for 50 timesteps and evaluated on predictions of closing prices n steps into the future. Input features were standardized to lie between 0 and 1. This preprocessing was done to ensure consistency in input scale across different stocks, which should facilitate learning and limit overfitting to individual stocks. Across 50 stocks, the dataset totaled 73789 training samples, 1958 validation samples, and 1966 test samples, with the splits defined to ensure no overlap between them.

The baselines were trained purely on predicting 3 or 10 steps into the future. To generate predictions of arbitrary length at evaluation time, we took model predictions of a single timestep, updated the 50-timestep input window with the prediction, and applied the baseline models autoregressively to sample the next timestep. In contrast, the latent SDEs and SRDMs were trained on both the ELBO (which includes a reconstruction loss and the KL loss) and the prediction loss. In addition, these models can be unrolled over arbitrary time windows, so no additional modifications were needed for varying-length predictions. We first discuss the impact of various SRDM design choices and then compare its performance to the baselines.

Effect of encoder context and posterior parametrization scheme. As described in Section 3.2, the posterior drift function can be parametrized in three different ways, and forecasting can be done using either the prior or posterior drift functions, resulting in 5 model variants. We report the performance of these variants when trained on 3-step forecasting in Table 1.

We find that using "full" context yields significantly better performance than the alternative context schemes. We also observe that forecasting with the posterior drift function performs slightly better than with the prior. However, we note that the performance gaps there are minimal, potentially indicating that the learned prior drift functions accurately capture typical market dynamics. Based on these results, for all other experiments in the paper we use SRDM with "full" context and forecasting with the posterior drift function.

Effect of Monte Carlo sample size. The training of SRDMs relies on a Monte Carlo approximation to the KL term in the ELBO. On each forward pass, one can sample multiple trajectories from the

context type	forecast dynamics	1-step R^2	3-step R^2	10-step R^2
full	prior	0.95451	0.92053	0.79373
full	posterior	**0.96053**	**0.93073**	**0.81710**
constant	prior	0.78642	0.75565	0.64642
constant	posterior	0.80553	0.77948	0.68535
IC-only	prior	0.81044	0.78275	0.67598

Table 1: Performance of SRDM model variants on 1, 3, and 10-step forecasting.

posterior, which may potentially reduce variance in gradients. As we show in Table 2, performance does somewhat improve as the number of samples increases. However, the benefit is relatively marginal compared to the increase in training time. As a result, for all other experiments in the paper, we simply use 1 posterior sample in training.

posterior samples	1-step R^2	3-step R^2	10-step R^2
1	0.96078	0.92922	0.79702
4	0.96130	0.92556	0.75737
8	0.96227	**0.93110**	**0.81132**
16	**0.96398**	0.92764	0.78549

Table 2: Performance of SRDM using varying numbers of posterior samples per forward pass.

Comparison to baseline methods. Having established the most performant model variant, we now compare its performance to various baseline methods in Table 3. We find that SRDM is highly competitive with conventional baselines, with SRDM trained on 3-step prediction ranking 1st or 2nd across all prediction windows. In particular, alongside latent SDE, our model shows strong performance when forecasting over windows longer than those it was trained on. We also note that for most models, performance degrades when training on longer-horizon prediction tasks. However, SRDM is robust to this, achieving near-identical performance whether it is trained on 3- or 10-step prediction.

Model	Training task	1-step R^2	3-step R^2	10-step R^2
GRU	3-step	**0.96229**	0.91653	0.77061
	10-step	0.93790	0.83802	0.43718
LSTM	3-step	0.95962	0.90180	0.67747
	10-step	0.95096	0.88639	0.62779
Transformer	3-step	0.95261	0.89519	0.66153
	10-step	0.84350	0.63563	0.11652
Latent SDE	3-step	0.94241	0.91297	**0.81061**
	10-step	0.93811	0.90090	0.73394
SRDM (ours)	3-step	<u>0.96137</u>	**0.92996**	<u>0.79988</u>
	10-step	0.95667	<u>0.92563</u>	0.79681

Table 3: Performance of SRDM and baseline models trained on 3- or 10-step forecasting and evaluated on 1, 3, and 10-step forecasting.

4.2 Market volatility analysis

We further analyze our trained models to explore whether and how market volatility is captured by their learned dynamics. Market volatility is typically measured as the standard deviation of returns over a specified period. We explore several features that could encode market volatility. First, we measure the average magnitude of the inferred diffusion term $g_\psi(z_t)$ across stocks for each day. Second, we compute the determinant of the Jacobian of the posterior drift $h_{\phi(x)}(z_t)$, also averaged across stocks for each day, as a measure of the local variability of the deterministic drift dynamics. Finally, we take the average prediction error (MSE) across all stocks for each day as a measure of the unpredictability of price fluctuations.

As shown in Figure 1, the inferred diffusion term shows the best match to a standard measure of market volatility, the Chicago Board Options Exchange's CBOE Volatility Index (VIX). It achieves a correlation of around 0.7 with VIX and clearly matches many of its fluctuations. In contrast, both the Jacobian determinant of the drift and the prediction error show very little correspondence with VIX. In particular, the prediction error, which is the only of these features that is available to typical timeseries forecasting models, has the poorest match to market volatility.

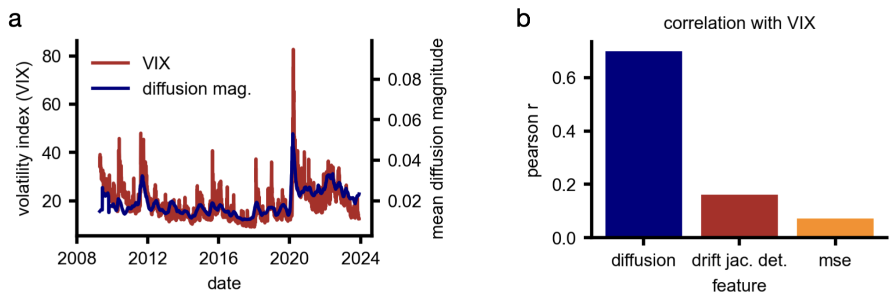

Figure 1: Correspondence between model features and market volatility. a) Average magnitude of the stochastic diffusion inferred by SRDM plotted against a standard market volatility index (VIX). b) Correlation between VIX and model features: "diffusion" is the average magnitude of the diffusion term inferred by SRDM; "drift jac. det." is the determinant of the Jacobian of the drift term inferred by SRDM; and "mse" is the mean squared error between SRDM forecasting predictions and true stock prices.

5 Discussion

In this work, we applied stochastic dynamical models to financial data and evaluated their utility for forecasting and analysis. We found that our stochastic dynamical models are competitive with deterministic baselines and outperform them on longer-horizon prediction tasks. In addition, stochastic models are more robust to training on longer-horizon prediction tasks, which degraded baseline model performance. This is potentially due to their decoupling of structured deterministic dynamics and stochastic variability in their drift and diffusion functions. Finally, we found that the model-inferred diffusion terms correlated well with measures of overall market volatility. This suggests that the learned dynamics are meaningful and potentially useful for gaining insight into market dynamics. Overall, our results suggest that explicitly modeling stochastic dynamics, and in particular independently modeling deterministic and stochastic components of the dynamics, may be beneficial for forecasting and data analysis.

However, we note that our model evaluations are relatively limited, relying on single runs of each model configuration and lacking baselines more specialized for timeseries forecasting. We also acknowledge that our baseline results differ substantially from those reported in [18]. However, our evaluation differs from theirs—when investigating their code, we found many inconsistencies in their data processing and evaluation across their baseline methods, so we instead opted to implement our own experimental setup. For our SRDM model, we also did not thoroughly investigate the impact of many design choices. In particular, it would be informative to ablate the minor differences that set SRDM apart from existing probabilistic SSMs and see if they contribute to performance, have no effect, or are even detrimental.

In all, our work provides preliminary evidence that neural network-based stochastic models can be well suited to modeling financial data, which is highly variable and unpredictable. This opens the door for future projects to more thoroughly explore what insights can be extracted from trained models and how they can be used to inform trading decisions or guide market interventions and economic policy.

Code availability

All code used to implement models and evaluations and run experiments can be found at https://github.com/felixp8/sde-finance.

References

[1] 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.

[2] Bruno Dupire et al. Pricing with a smile. *Risk*, 7(1):18–20, 1994.

[3] Christian Naesseth, Scott Linderman, Rajesh Ranganath, and David Blei. Variational sequential monte carlo. In *International conference on artificial intelligence and statistics*, pages 968–977. PMLR, 2018.

[4] Hanyao Gao, Gang Kou, Haiming Liang, Hengjie Zhang, Xiangrui Chao, Cong-Cong Li, and Yucheng Dong. Machine learning in business and finance: a literature review and research opportunities. *Financial Innovation*, 10(1):86, 2024.

[5] 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.

[6] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.

[7] Abdul Fatir Ansari, Lorenzo Stella, 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 Yuyang Wang. Chronos: Learning the language of time series, 2024.

[8] Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers, 2024.

[9] Yifan Hu, Yuante Li, Peiyuan Liu, Yuxia Zhu, Naiqi Li, Tao Dai, Shu tao Xia, Dawei Cheng, and Changjun Jiang. Fintsb: A comprehensive and practical benchmark for financial time series forecasting, 2025.

[10] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.

[11] Saurabh Vyas, Matthew D Golub, David Sussillo, and Krishna V Shenoy. Computation through neural population dynamics. *Annual review of neuroscience*, 43(1):249–275, 2020.

[12] Matthijs Pals, A Erdem Sağtekin, Felix Pei, Manuel Gloeckler, and Jakob H Macke. Inferring stochastic low-rank recurrent neural networks from neural data. *arXiv preprint arXiv:2406.16749*, 2024.

[13] Andreas Doerr, Christian Daniel, Martin Schiegg, Nguyen-Tuong Duy, Stefan Schaal, Marc Toussaint, and Trimpe Sebastian. Probabilistic recurrent state-space models. In *International conference on machine learning*, pages 1280–1289. PMLR, 2018.

[14] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR, 2019.

[15] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations, 2019.

[16] Xuanqing Liu, Tesi Xiao, Si Si, Qin Cao, Sanjiv Kumar, and Cho-Jui Hsieh. Neural sde: Stabilizing neural ode networks with stochastic noise. *arXiv preprint arXiv:1906.02355*, 2019.

[17] Xuechen Li, Ting-Kam Leonard Wong, Ricky T. Q. Chen, and David Duvenaud. Scalable gradients for stochastic differential equations, 2020.

[18] Zihan Dong, Xinyu Fan, and Zhiyuan Peng. Fnspid: A comprehensive financial news dataset in time series. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4918–4927, 2024.

[19] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960.

[20] Scott W Linderman, Andrew C Miller, Ryan P Adams, David M Blei, Liam Paninski, and Matthew J Johnson. Recurrent switching linear dynamical systems. *arXiv preprint arXiv:1610.08466*, 2016.

[21] Patrick Kidger, James Foster, Xuechen Li, and Terry J Lyons. Neural sdes as infinite-dimensional gans. In *International conference on machine learning*, pages 5453–5463. PMLR, 2021.

[22] David Sussillo, Rafal Jozefowicz, LF Abbott, and Chethan Pandarinath. Lfads-latent factor analysis via dynamical systems. *arXiv preprint arXiv:1608.06315*, 2016.

A Appendix

A.1 Model architectures

Here we provide graphical depictions of our SRDM architectural variants. We depict only the inference process and thus do not show the prior dynamics model.

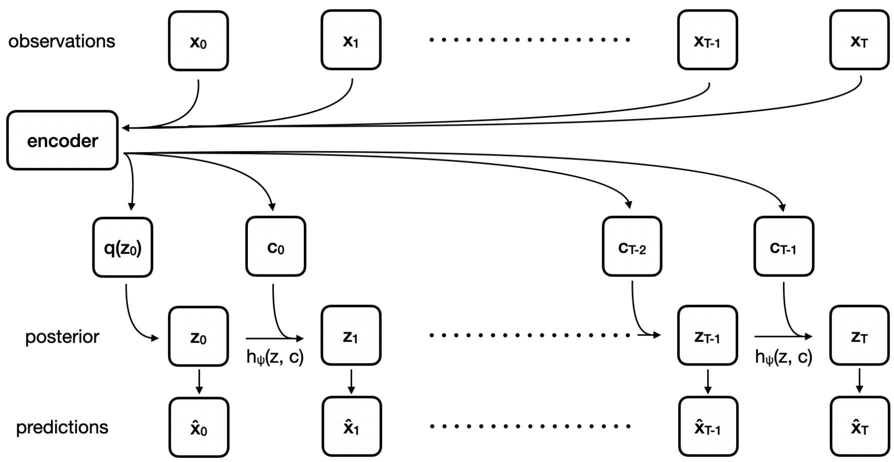

Figure 2: The "full" context variant of SRDM, where the encoder provides a time-varying context vector to the posterior drift at each timestep.

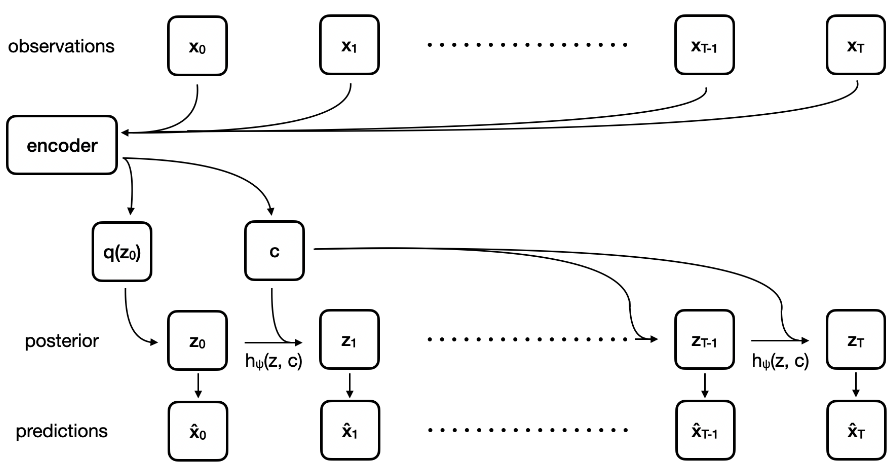

Figure 3: The "constant" context variant of SRDM, where the encoder outputs a single context vector that is repeated at each timestep for the posterior drift function.

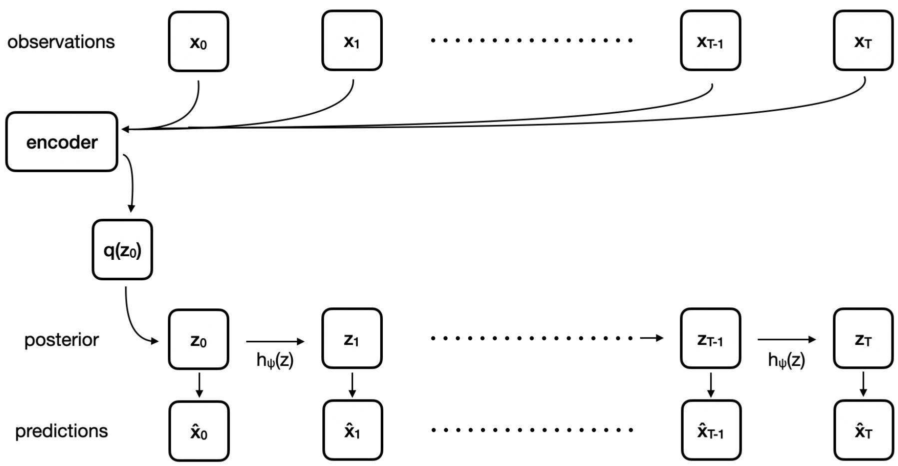

Figure 4: The "IC-only" context variant of SRDM, where the encoder does not provide a context vector to the posterior drift function and only sets the initial condition.