ON AUTOREGRESSIVE TIME SERIES GENERATION USING FLOW-BASED GENERATIVE MODELS

Edmond Cunningham
University of Massachusetts Amherst

Madalina Fiterau
University of Massachusetts Amherst

Daniel Sheldon
University of Massachusetts Amherst

ABSTRACT

Flow-based generative models (FBGM) have emerged as a dominant approach to generative modeling in many domains for their scalability and controllability, but have notably not made the same impact on autoregressive probabilistic forecasting. Although the methodology behind these models can be applied directly to the time series setting, and in theory offers the potential to apply the advances in generative modeling to time series, this direct approach has only recently been explored. We investigate this methodological gap by generalizing the key elements of flow-based generative modeling to the autoregressive, time series setting. We show that FBGMs, based on linear stochastic differential equations, are solutions to a general mean-field variational inference (MFVI) problem for conditional exponential family distributions. Solutions to this MFVI problem are available in closed form, learnable via mean-squared error minimization, and encapsulate familiar time series models including mean-squared error based forecasters, Gaussian autoregressive models and a discrete time extension of the SDE used in FBGMs. We offer deep insights into the construction of FBGMs, with strong connections to traditional time series models, as well as a high performance software package for linear SDE based FBGMs.

1 Introduction

Flow-based generative models (FBGM), including denoising diffusion, score based diffusion, and flow matching models, have become the dominant approach to generative modeling. These models represent a stochastic differential equation (SDE) that transforms samples from a known prior distribution into samples from an unknown target distribution, and often use a different recipe for solving the generative modeling problem compared to traditional approaches. This alternative approach is highly scalable [Ramesh et al., 2022, Podell et al., 2023, Saharia et al., 2022], can leverage conditioning information in flexible ways [Dhariwal and Nichol, 2021, Ho and Salimans, 2022], and can be controlled in order to incorporate user defined dynamics [Liu et al., 2024, Domingo-Enrich et al., 2024, Havens et al., 2025]. Furthermore, FBGMs are capable of learning from paired data. If x_0 and x_1 are samples from an unknown joint distribution $p(x_0, x_1)$, then one can use the same approach to construct an SDE whose transition distribution from $t = 0$ to $t = 1$ is $p(x_1|x_0)$ [De Bortoli et al., 2023]. Given this capability, it directly follows that this approach could, in principle, be used to construct an SDE to model time series data. If $p(x_{1:N}) = p(x_1)\prod_{k=1}^{N-1} p(x_{k+1}|x_{1:k})$ represents the unknown distribution of time series data, then each of the transition terms, $p(x_{k+1}|x_{1:k})$, can be interpreted as a target distribution for a FBGM in the paired data setting where the data pairs are consecutive elements of the time series, (x_{k+1}, x_k), and the previous elements $x_{1:k-1}$ can be thought of as extra conditioning information. In theory, learning this kind of model for time series would inherit the scalability and controllability that FBGMs possess, allowing practitioners to port over the recent advances in generative modeling to time series applications. However, this approach has surprisingly only recently been explored [Chen et al., 2024a, Tamir et al., 2024, Park et al., 2024, Chen et al., 2024b] even though diffusion based time series models have been studied for several years [Yang et al., 2024, Meijer and Chen, 2024]. To address this gap, we develop a discretized time series model that is founded on the same theoretical principles as FBGMs, while being substantially easier to work with in practice. We do this by generalizing two key elements needed to construct FBGMs, stochastic interpolation and the Markovian projection, to the time series setting, where they become Gaussian conditional random fields and a form of mean-field variational inference respectively.

We construct a family of latent probabilistic time series models that are closely related to existing time series models, including MSE based non-probabilistic forecasters and conditional Gaussian autoregressive models, and compare their performance on various latent probabilistic forecasting problems.

2 Background

We will first review how flow-based generative models are constructed and then build intuition for how to go about generalizing this construction to the time series setting. Suppose that $p(y_0, y_1)$ is a joint distribution over a source and target random variable. The (paired) generative modeling problem is to find a parametric approximation of $p(y_1|y_0)$ [1]. Flow-based generative models solve this problem by constructing, and then learning, a *latent* SDE whose transition distribution from times $t = 0$ to $t = 1$ is $p(y_1|y_0)$. There are three steps involved in constructing and learning this SDE - **stochastic interpolation**, the **Markovian projection**, and **matching**.

Stochastic interpolation [Albergo and Vanden-Eijnden, 2023] is used to interpolate between probability distributions by interpolating their samples. For example, consider the joint distribution $p(x_0, x_t, x_1)$, where $x_t = (1-t)x_0 + tx_1$ and $(x_0, x_1) \sim p(x_0, x_1)$. By the definition of x_t, it is true that $p(x_{t=1}) = p(x_1)$, and also that $p(x_{t=1}|x_0) = p(x_1|x_0)$, so we verify that the marginal distribution of x_t interpolates between $p(x_0)$ and $p(x_1)$. In practice, one assumes that at times $t = 0$ and $t = 1$, $x_0 := y_0$ and $x_1 := y_1$ so that $p(x_t)$ is an interpolation between $p(y_0)$ and $p(y_1)$.

A popular method for constructing stochastic interpolants is conditioning a user-defined base SDE, whose diffusion coefficient does not depend on the current state, to start at x_0 and end at x_1. This SDE takes the form $dx_t = b_t(x_t)dt + L_t dW_t$ where $b_t(x_t)$ is the drift of this base SDE and L_t is the diffusion coefficient. This SDE is used to construct a joint distribution of the form $p(x_0, x_t, x_1) = p(x_t|x_0, x_1)p(x_0, x_1)$ where $p(x_t|x_0, x_1)$ is the probability of x_t when the base SDE has been conditioned to start at x_0 and end at x_1. In order to solve the generative modeling problem, the marginal distribution of FBGMs is constructed to be $p(x_t|x_0)$ using the **Markovian projection**.

Proposition 1 (Markovian projection SDE [Shi et al., 2024])**.** *Let $p(x_1|x_0)$ be a conditional distribution and let $p(x_t|x_0, x_1)$ denote the distribution of a sample from the base SDE $dx_t = b_t(x_t)dt + L_t dW_t$ that is conditioned to start at x_0 and end at x_1. The "Markovian projection SDE" is an SDE whose marginal distribution, denoted by $q^*(x_t|x_0)$ is equal to $p(x_t|x_0)$. It is given by:*

$$dx_t = \big(b_t(x_t) + L_t L_t^T \, s_t^*(x_t, x_0)\big)dt + L_t dW_t \quad \textit{where } s_t^*(x_t, x_0) = \mathbb{E}_{p(x_1|x_0,x_t)}[\nabla \log p(x_1|x_0, x_t)]$$

See Prop 3. of [De Bortoli et al., 2023] for a proof. Proposition 1 is a solution to the paired generative modeling problem because $q^*(x_{t=1}|x_0) = p(x_1|x_0) := p(y_1|y_0)$. Given a sample from the source distribution, $x_0 \sim p(x_0)$, we can simulate the SDE from $t = 0$ to $t = 1$ to generate a sample from the target distribution. However, this SDE contains the intractable term $s_t^*(x_t, x_0)$. This term is learned using a **matching** algorithm, such as score matching [Vincent, 2011, Song et al., 2021], whose objective looks as follows:

$$\nabla \log q^*(x_t|x_0) = \underset{s_t(x_t,x_0)}{\operatorname{argmin}} \, \mathbb{E}_{p(x_0,x_1,x_t)} \left[\left\| L_t L_t^T \nabla \log p(x_1|x_0, x_t) - s_t(x_t, x_0) \right\|^2 \right] \tag{1}$$

If $s(x_t, x_0; \theta)$ is parameterized by a neural network, then one can minimize this expectation with standard techniques. After training is complete, then the flow-based generative model is given by the SDE $dx_t = (b_t(x_t) + L_t L_t^T s_t(x_t, x_0))dt + L_t dW_t$. In general, matching algorithms, such as score matching, drift matching and bridge matching, are algorithms for learning the Bayes estimator of a random variable because of the well known relationship between posterior expectations and mean squared error [Jaynes, 2003]:

Proposition 2 (Bayes estimate of parameter)**.** *Let $p(z, \theta)$ be a joint distribution and let $\theta^*(z)$ be the Bayes estimate of θ based on z under the squared error risk. Then the Bayes estimate takes the following two forms:*

$$\theta^*(z) = \mathbb{E}_{p(\theta|z)}[\theta] = \underset{f(z)}{\operatorname{argmin}} \; \mathbb{E}_{p(z,\theta)} \left[\|f(z) - \theta\|^2 \right] \tag{2}$$

See Section 9.3 for a derivation. In score matching, one would have $z = (x_0, x_t)$ and $\theta = \nabla \log p(x_1|x_0, x_t)$, while other matching approaches, such as flow matching [Albergo and Vanden-Eijnden, 2023, Lipman et al., 2023, Liu et al., 2023] and bridge matching [Shi et al., 2024].

Given that FBGMs possess strong theoretical and empirical properties of FBGM for generative modeling, the goal of our paper is to see how one can port over these properties to time-series forecasting. To do this, we will identify the core modeling and learning principles that drive flow-based generative models and re-interpret them from a graphical models perspective to directly apply them to time-series forecasting.

[1] The unpaired setting is when we do not condition on y_0.

3 Related Work

There are numerous perspectives on flow-based generative models [Luo, 2022, Dieleman, 2023] and even more variants of these models. At their core, these models start by constructing a stochastic process that starts at a prior distribution and ends at the data distribution. Diffusion models use progressive noising of data to build this map [Sohl-Dickstein et al., 2015, Ho et al., 2020, Song et al., 2021] via a simple SDE whose stationary distribution is Gaussian. On the other hand, flow-matching models [Liu et al., 2023, Albergo and Vanden-Eijnden, 2023, Lipman et al., 2023] use a stochastic bridge to build this map by conditioning a simple SDE to start at a point in the prior distribution and end at the data distribution. The choice of simple SDE used in all of these models is a user-defined choice that typically is a linear SDE, such as variance preserving SDE [Song et al., 2021], Brownian motion, Ornstein-Uhlenbeck process, and others, due to their tractability as Gaussian processes [Särkkä and Solin, 2019], and is even used to construct more exotic latent SDEs such as critically damped langevin dynamics [Dockhorn et al., 2022, Chen et al., 2024c] or the Weiner velocity model [Bar-Shalom et al., 2001, Särkkä et al., 2006]. In our paper, we abstract away these choices and generally consider using linear SDEs to construct the initial map between distributions. There are a few different ways to go from this initial stochastic process to a FBGM. A common way to construct a FBGM from this is construct and optimize an ELBO for the likelihood of data under this initial process [Kingma et al., 2021]. Alternatively, one can directly solve for the SDE whose marginal distribution is that of this initial process [Song et al., 2021, Lipman et al., 2023] or define it as the SDE whose path measure is as close as possible to the initial process [Shi et al., 2024, De Bortoli et al., 2023] in terms of KL divergence, called the Markovian projection. We adopt the latter view over the ELBO view because it explicitly constructs a solution to the generative modeling problem and is available in closed form while this is hidden in the ELBO formulation and show that the solution to a mean field variational inference problem can be seen as an approximate discrete time counterpart.

Flow-based generative models have been successfully applied to time series problems in a *non-autoregressive* fashion [Kollovieh et al., 2023, Yuan and Qiao, 2024, Kollovieh et al., 2025, Hu et al., 2024, Yang et al., 2024, Meijer and Chen, 2024]. These models transform the time series generative modeling problem into the standard generative modeling problem used in image generation by treating each time series as a single vector by concatenating all times together, and then learning a map from a Gaussian vector of the same size to the data vector. These approaches can be conditioned using guidance [Rasul et al., 2021, Dhariwal and Nichol, 2021, Ho and Salimans, 2022, Kollovieh et al., 2023] which allows them to perform tasks such as forecasting and imputation. Our approach differs from these in that we construct autoregressive models.

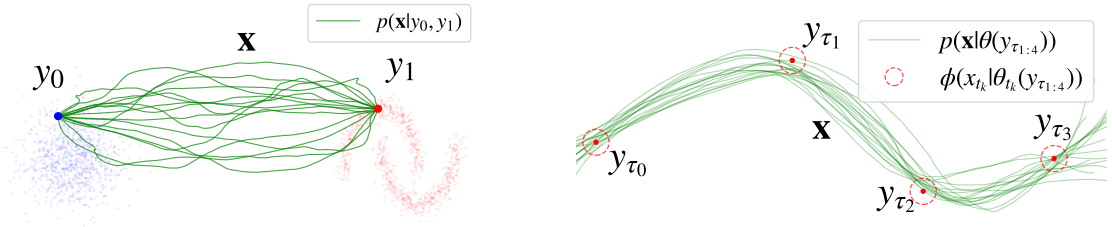
The class of models most relevant to our paper are autoregressive neural SDEs that are trained using principles from flow-based generative models. [Chen et al., 2024a] uses a Follmer process to model the transition distributions of the distribution of time series data, which is the same approach that we adopt in our Neural SDE model. [Park et al., 2024] also learns a similar latent Neural SDE model that uses a similar form of soft conditioning as us (through the use of emission potentials), and is trained to maximize the likelihood of data. [Tamir et al., 2024] is also similar where they perform stochastic interpolation using Gaussian processes and perform inference with Kalman smoothing as well, which is a form of message passing. Finally, [Shen and Cheng, 2025] learns a more general SDE to learn the distribution of time series data where the diffusion coefficient is not independent of the current state and also maximize the likelihood of data. These related papers are all related to the Neural SDE that we describe in our paper. Our main contributions are centered around investigating how to apply the approach used to construct these continuous time models for creating similar discrete time models. [El-Gazzar and van Gerven, 2025] used flow matching to learn the next state distribution of time series data, but did not learn a Föllmer process for this task and instead learned to transform a Gaussian into the next state distribution.

4 Generalized linear stochastic interpolation

Let $y_{\tau_{1:T}} \sim p(y_{\tau_{1:T}})$ denote time series data and let $\mathbf{x}$ be a sample from a linear SDE. Generalized linear stochastic interpolation is about constructing $p(\mathbf{x}|y_{\tau_{1:T}})$, for which we will later use to build and perform inference in the forecasting distribution $p(\mathbf{x}|y_{\tau_{1:K}})$, where $K < T$. Throughout this paper, we will use the notation $\phi(x|\theta)$ to denote an unnormalized multivariate Gaussian distribution over x with natural parameters θ. See Section 9 for a review of the natural parameter form of Gaussians. We also use the notation $\phi_{k+1|k}(x_{k+1}|x_k) = N(x_{k+1}|Ax_k + u, \Sigma)$ to denote a Gaussian transition distribution from x_k to x_{k+1} with state transition matrix A, bias vector u and covariance matrix Σ.

4.0.1 Gaussian conditional random fields

Chain structured Gaussian CRFs are a tractable class of probabilistic models defined as follows:



(a) Stochastic interpolation

(b) Generalized stochastic interpolation

Figure 1: Generalized stochastic interpolation incorporates Gaussian potential functions to relax the endpoint conditions of stochastic interpolation and is applied to time series data.

Definition 1 (Conditional Random Field [Lafferty et al., 2001, Sutton et al., 2012])**.** *Let $x_{1:N}$ be a sequence of random variables, $\phi_{k+1|k}(x_{k+1}|x_k)$ be a set of Gaussian transition distributions between consecutive variables, and $\phi(x_k|\theta_k)$ a fixed set of Gaussian potential functions with natural parameters $\theta_k \in \theta$. A conditional random field (CRF) is a probability distribution given by:*

$$p(x_{1:N}|\theta) \propto \prod_{k=1}^{N-1} \phi_{k+1|k}(x_{k+1}|x_k) \prod_{k=1}^{N} \phi(x_k|\theta_k) \tag{3}$$

Due to the chain-structure of $p(x_{1:N}|\theta)$ and the fact it is jointly Gaussian, inference can be performed efficiently using message passing. The backward messages, defined below, play a significant role in our theory:

Proposition 3 (Backward messages)**.** *The k'th backward message of a CRF is defined by the following recurrence relation:*

$$\phi(x_{k-1}|\beta_{k-1}) = \int \phi_{k|k-1}(x_k|x_{k-1})\phi(x_k|\theta_k + \beta_k)dx_k \tag{4}$$

where $\theta_{k+1} + \beta_{k+1}$ denotes the direct sum of θ_{k+1} and β_{k+1} and $\beta_N = 0$. This recurrence can also be expressed through the function $\Phi_{k-1,k}$ that performs the parameter updates as:

$$\beta_{k-1} = \Phi_{k-1,k}(\theta_k + \beta_k) \tag{5}$$

See Definition 6 for its precise form.

See Section 10 for a full derivation of sequential and parallel message passing, and Section 14 for pseudo code and implementation considerations. Although we do not focus on the forward messages, they are defined with analogous recurrence relations to the backward messages and can be used to extend our methodology to flow-matching models for time series forecasting (see Corollary 6). CRFs offer an efficient way to model the latent variables at a fixed set of times, but are not immediately suited for continuous time. Extending CRFs to coninuous time simply requires choosing transitions that correspond to a continuous time stochastic process.

4.0.2 Linear stochastic differential equations

We use linear SDEs to construct the transition distributions of continuous time CRFs. Linear SDEs have the form $dx_t = (F_t x_t + u_t)\,dt + L_t dW_t$, where the drift matrix F_t, bias vector u_t and diffusion coefficient matrix L_t do not depend on the current state x_t, and have the convenient property that their transition distribution is Gaussian and easy to compute [Särkkä and Solin, 2019, Singhal et al., 2023]. The next proposition shows that a CRF that is constructed with linear SDE transitions corresponds exactly to the joint distribution of a conditioned linear SDE that is discretized at a given set of times:

Proposition 4 (Conditioned LTI-SDE)**.** *Let $\phi_{t+s|t}(x_{t+s}|x_t)$ be the transition distribution of the linear SDE $dx_t = (F_t x_t + u_t)\,dt + L_t dW_t$ and let $\{\phi(x_{t_k}|\theta_{t_k})\}_{t_k \in \mathcal{R}}$ be potential functions at times in the set $\mathcal{R}$. Then the piecewise-linear SDE,*

$$dx_t = (F_t x_t + u_t + L_t L_t^T \nabla \log \phi(x_t|\beta_t))dt + L_t dW_t, \tag{6}$$

$$x_{t_1} \sim \phi(x_{t_1}|\beta_1 + \theta_1)$$

where $t \in (t_k, t_{k+1})$ and $t_k, t_{k+1} \in \mathcal{R}$, has a joint distribution at the times $t_{1:N} = \mathcal{T} \supseteq \mathcal{R}$ that is given by a CRF:

$$p(x_{t_{1:N}}|\theta) \propto \prod_{t_k \in \mathcal{T}} \phi_{t_{k+1}|t_k}(x_{t_{k+1}}|x_{t_k}) \prod_{t_k \in \mathcal{R}} \phi(x_{t_k}|\theta_{t_k}) \tag{7}$$

where $\beta_t = \Phi_{t,t_{k+1}}(\theta_{t_{k+1}} + \beta_{t_{k+1}})$.

See appendix Section 11.1 for the full proof and Corollary 6 for a nice expression for the associated probability flow ODE in terms of both the forward and backward messages. Proposition 4 offers an explicit construction for $p(\mathbf{x}|y_{\tau_{1:T}})$. If we simply let each parameter θ_{t_k} depend on a single obervation y_{t_k}, then Proposition 4 gives us a model for $p(\mathbf{x}|\theta) = p(\mathbf{x}|y_{\tau_{1:T}})$. Additionally, we can perform inference in this distribution efficiently by discretizing a CRF and using parallel message passing algorithms to perform inference in $O(\log |\mathcal{T}|)$ time [Hassan et al., 2021, Corenflos et al., 2021, Smith et al., 2023]. Next we will show how generalized stochastic interpolation can be used to define an inference problem that corresponds to probabilistic forecasting.

5 Probabilistic forecasting with generalized linear stochastic interpolants

Let $p(\mathbf{x}, y_{\tau_{1:T}}) = p(y_{\tau_{1:T}})p(\mathbf{x}|y_{\tau_{1:T}})$ be the joint distribution of data and the latent process constructed in Proposition 4. Suppose we split each sequence of data into observed and unobserved portions, $y_{\tau_{1:T}} = (y_{\mathcal{O}}, y_{\mathcal{U}})$, where $y_{\mathcal{O}}$ is a subsequence that we observe at both train and test time while $y_{\mathcal{U}}$ is only observed at training time, as is the case in time series forecasting.[2] Although probabilistic forecasting typically corresponds to performing inference in the distribution $p(y_{\mathcal{U}}|y_{\mathcal{O}})$, we consider the more general problem of performing inference in $p(\mathbf{x}|y_{\mathcal{U}})$. It is easy to see that in the special case where the Gaussian potential functions for $p(\mathbf{x}|y_{\tau_{1:T}})$ are chosen as dirac delta functions - $\phi(x_{t_k}|\theta_{t_k}(y_{\tau_{1:T}})) := \delta(x_{t_k} - y_{t_k})$, then the two distributions agree $p(x_{\mathcal{U}}|y_{\mathcal{O}}) = p(y_{\mathcal{U}}|y_{\mathcal{O}})$. As such, the goal of this section is to devise variational inference algorithms for $p(\mathbf{x}|y_{\mathcal{O}})$.

5.1 Neural latent SDE

We start by introducing the direct extension of the neural SDEs learned in FBGMs translates to the time series setting. If we simply treat consecutive latent variables $(x_{t_k}, x_{t_{k+1}})$ as elements of a paired dataset with the previous elements $x_{t_{1:k-1}}$ and observations $y_{\mathcal{O}}$ as extra conditioning information, then we can construct a piecewise Markovian projection SDE Proposition 1 to model the distribution of time series data:

Proposition 5 (Neural latent SDE)**.** *Let $p(x_{t_{1:N}}, y_{\tau_{1:T}})$ be the joint distribution of data and a linear stochastic interpolant and suppose that $y_{\tau_{1:T}} = (y_{\mathcal{O}}, y_{\mathcal{U}})$, where $\mathcal{O}$ and $\mathcal{U}$ are the times at which sequences are observed and unobserved at test time, respectively. Then the neural latent SDE is the following piecewise SDE:*

$$dx_t = \left(F_t x_t + u_t + L_t L_t^T \nabla \log \phi(x_t|\beta_t^*(x_t, x_{t_{1:k}}, y_{\mathcal{O}}))\right) dt + L_t dW_t, \tag{8}$$
$$\text{where } \beta_t^*(x_t, x_{t_{1:k}}, y_{\mathcal{O}}) = \mathbb{E}_{p(y_{\mathcal{U}}|x_t, x_{t_{1:k}}, y_{\mathcal{O}})}\left[\beta_t(y_{\tau_{1:T}})\right], \text{ and } t \in (t_k, t_{k+1})$$

Furthermore, the transition distribution of this SDE from time t_k to t_{k+1} is $p(x_{t_{k+1}}|x_{t_{1:k}}, y_{\mathcal{O}})$. We will use $q^{\text{Neural-SDE}}$ to denote the path measure associated to this SDE.

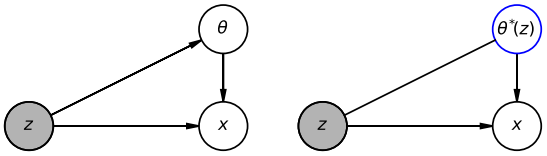
See Section 13.2 for a proof and Section 13 for the general constructions of the score function, Markovian projection SDE and probability flow ODE. By construction, Proposition 5 can be used to solve the latent probabilistic forecasting problem because it has the correct joint distribution over the latent space. Furthermore, its form is almost identical to that of its base LTI-SDE in Proposition 4, except that its parameter, β^*, is the Bayes estimator of a backward message.

The problem with this model is that compounding numerical errors during simulation make it unreliable and slow to use in practice. As we will argue later, this class of models may be overkill for the generative modeling of time series because one might expect that the transition distribution between consecutive times in the data may not require a complicated distribution to model. This is in contrast to the image generative modeling setting where the transition from a Gaussian to the distribution of image data is complex. Motivated by these observation, we investigate the core algorithmic principles behind FBGMs in the next section to devise an algorithm more suited for time series modeling.

5.2 Constrained mean-field variational inference

Next we introduce our main contribution which is the variational inference algorithm underlying FBGMs, which we call "constrained mean-field variational inference". Given a conditional exponential family distribution $p(x|z, \theta)$, CMFVI

[2]This also covers the imputation setting, but we do not explore this in this paper.



(a) $p(x|z,\theta)$ (b) $q^*(x|z) = p(x|z,\theta^*(z))$

Figure 2: The CMFVI approximation of $p(x|z)$ is $q^*(x|z)$. Choosing $(x,z,\theta) = (x_{t_{1:N}}, y_{\mathcal{O}}, \theta(y_{\tau_{1:T}}))$ recovers q^{MSE}, $(x,z,\theta) = (x_{t_k}, (x_{t_{1:k-1}}, y_{\mathcal{O}}), \theta(y_{\tau_{1:T}}))$ recovers $q^{\text{MSE-AR}}$ and $(x,z,\theta) = \lim_{s\to 0}(x_{t+s}, (x_t, x_{t_{1:k-1}}, y_{\mathcal{O}}), \theta(y_{\tau_{1:T}}))$ for $t \in (t_k, t_{k+1})$ recovers $q^{\text{Neural-SDE}}$.

constructs a variational approximation of $p(x|z)$ that is given by $p(x|z,\theta^*(z))$ where $\theta^*(z)$ is the Bayes estimator of θ given z. We first introduce CMFVI in an abstract way and then show how it can be used to do variational inference for $p(x_{t_{1:N}}|y_{\mathcal{O}})$.

Suppose that z is a random variable, $\theta \sim p(\theta|z)$ is the natural parameter of an exponential family distribution, and $x \sim p(x|z,\theta)$ is a random variable drawn from a conditional exponential family of the form $p(x|z,\theta) = \exp\{\langle t_z(x), \theta\rangle - A(z,\theta)\}$. For intuition, assume that x represents the future of a stochastic process, z represents its past , and θ represents the parameters of this process. Furthermore, suppose that θ is only available at training time so that at test time, sampling x given z requires the ability to sample from $p(x|z)$. Sampling from $p(x|z)$ is difficult because $p(\theta|z)$ is arbitrary, and so we resort to variational inference:

Theorem 1 (Constrained mean field VI solution)**.** *Let $p(x|z,\theta) \propto \exp\{\langle t_z(x), \theta\rangle - A(z,\theta)\}$ be a conditional exponential family distribution with $\theta \sim p(\theta|z)$. The constrained mean field VI approximation of $p(x|z)$, denoted by $q^*(x|z)$, is defined as follows:*

$$q^*(x|z) = \operatorname*{argmin}_{q(x|z)} \mathrm{KL}\left[q(x|z)p(\theta|z) \| p(x,\theta|z)\right] = p(x|z,\theta^*(z)), \quad \textit{where } \theta^*(z) = \mathbb{E}_{p(\theta|z)}\left[\theta\right] \tag{9}$$

See Section 12.1 for a proof. The parameter $\theta^*(z)$ is the Bayes estimator of θ given z and by Proposition 2 can be learned using mean squared error minimization, provided that it is possible to sample from $p(z,\theta)$. While this variational approximation is tractable, it seems restrictive because it is a conditional random field and only exact when θ and x are conditionally independent given z. However, this may not be a terrible assumption in the time series setting. If the process is deterministic, then we should be able to compute x directly from z without needing to know θ, and so this independence assumption will hold because one will be able to compute the future values of the process directly from its past. Provided that the process underlying data is not too stochastic, we should expect that, given a long enough history and a short enough prediction horizon, CMFVI can yield a reasonable approximation of $p(x|z)$. This intuition motivates the use of CMFVI for learning the autoregressive factors of $p(x_{t_{1:N}}|y_{\mathcal{O}})$ to solve the probabilistic forecasting problem.

5.3 CMFVI-based models

Next we use CMFVI to construct probabilistic forecasters. To start, we will need to make the assumption that the covariances of the potential functions are independent of the values of $y_{\tau_{1:T}}$. This assumption holds when we use dirac delta potential functions, and also in the case where the CRF is constructed as a linear dynamical system with constant observation noise. This assumption allows us to parameterize our models in terms of the Gaussian means instead of backward messages. For example, $q^{\text{Neural SDE}}$ can be rewritten in a more interpretable form where the only unknown value is the mean of the next backward message:

Corollary 1 (Neural latent SDE using potentials with fixed covariances)**.** *Suppose the covariance matrices associated the potential functions are constant with respect to y. Then the SDE associated with $q^{\textit{Neural SDE}}$ is:*

$$dx_t = (F_t x_t + u_t + L_t L_t^T \nabla \log N(x_t|\mu_t^*, \Sigma_t))dt + L_t dW_t \tag{10}$$

where $t \in (t_{k-1}, t_k)$, $\mu_t^* = \mu_t^{\beta^*}(x_t, x_{t_{1:k-1}}, y_{\mathcal{O}})$ is the Bayes estimator for the mean of $\phi(x_t|\beta_t(y_{\tau_{1:T}}))$, $\Sigma_t = \Sigma_t^{\beta}$ is its covariance, and the gradient operator does not pass through μ_t^*.

The result follows directly from converting β_{t_k} from natural parameters to standard parameters of a Gaussian and the linear equivariance of the Bayes estimator Section 12.2. Note that by our assumption that the parameters of the potential

functions do not depend on $y_{\tau_{1:T}}$, Σ_t^β can be computed by performing message passing on $p(x_{t_{1:N}}|\varnothing_{\tau_{1:T}})$, where $\varnothing_{\tau_{1:T}}$ is an empty (or random) sequence sampled at the same times as $y_{\tau_{1:T}}$.

Next, we show that time series forecasting models that are trained to minimize the mean squared error between the ground truth future of a sequence and predictions from a neural network can be interpreted as a naive CMFVI based time series model:

Corollary 2 (MSE Forecaster)**.** *Suppose the covariance matrices associated the potential functions are constant with respect to y. Then the MSE-CMFVI solution, denoted by $q^{MSE}(x_{t_{1:N}})$ is the CMFVI approximation of $p(x_{t_{1:N}}|y_\mathcal{O})$ obtained by choosing $(x, z, \theta) = (x_{t_{1:N}}, y_\mathcal{O}, \theta(y_{\tau_{1:T}}))$. The forecasting distribution it defines is given by:*

$$q^{MSE}(x_{t_{1:N}}|y_\mathcal{O}) \propto \prod_{t_k \in \mathcal{T}} \phi_{t_{k+1}|t_k}(x_{t_{k+1}}|x_{t_k}) \prod_{t_k \in \mathcal{R}} N(x_{t_k}|\mu^*_{t_k}(y_\mathcal{O}), \Sigma_{t_k})$$

where $\mu^*_{t_k}(y_\mathcal{O}) = \mathbb{E}_{p(y_\mathcal{U}|y_\mathcal{O})}[\mu_{t_k}(y_{\tau_{1:T}})]$ is the Bayes estimate of μ_{t_k}, and $\phi(x_{t_k}|\theta_{t_k}(y_{\tau_{1:T}})) = N(x_{t_k}|\mu^*_{t_k}(y_{\tau_{1:T}}), \Sigma_{t_k})$.

If one learns the unknown parameter $\mu^*_{t_k}(y_\mathcal{O})$ using mean-squared error minimization via Proposition 2, then one recovers precisely the learning algorithm used in MSE based, non-probabilistic forecasters where one minimizes the mean squared error between the model's predictions of the future of a time series and the ground truth future.

5.4 Discrete time Markovian projection

Finally, we can use CMFVI to construct an autoregressive model whose transitions are the CMFVI approximation of the autoregressive factors of $p(x_{t_{1:T}}|y_\mathcal{O})$.

Proposition 6 (Autoregressive CMFVI solution)**.** *Let $p(x_{t_{1:N}}|y_\mathcal{O})$ be the target distribution, assume that the covariance matrices of its potential functions are constant with respect to y. The autoregressive model whose transitions are CMFVI solution, denoted by $q^{MSE\text{-}AR}$ is given by:*

$$q^{MSE\text{-}AR}(x_{t_{1:N}}|y_\mathcal{O}) \propto p(x_{t_1}|y_\mathcal{O}) \prod_{t_k \in \mathcal{T}} \phi_{t_k|t_{k-1}}(x_{t_k}|x_{t_{k-1}}) N(x_{t_k}|\mu^*_{t_k}, \Sigma_{t_k})$$

where $\mu^*_{t_k} = {\mu^\beta_{t_k}}^*(x_{t_{1:k-1}}, y_\mathcal{O})$ and $\Sigma_{t_k} = \Sigma^\beta_{t_k}$ are the same as in Corollary 1. Furthermore, $q^{MSE\text{-}AR}$ has the same joint distribution over $x_{t_{1:N}}$ as the following piecewise linear SDE:

$$dx_t = (F_t x_t + L_t L_t^T \nabla \log N(x_t|\mu^*_t, \Sigma_t))dt + L_t dW_t, \quad x_{t_1} \sim p(x_{t_1}|y_\mathcal{O})$$

where $\mu^*_t = \mu^*_t(x_{t_{1:k-1}}, y_\mathcal{O})$ is the Bayes estimator for the mean of $\beta_t(y_{\tau_{1:T}}) = \Phi_{t,t_k}(\beta_{t_{k+1}}(y_{\tau_{1:T}}))$, $\Sigma_t = \Sigma^\beta_t$ is its covariance matrix and $t \in (t_{k-1}, t_k)$ for $k = 2, \ldots, T$.

See Section 12.3 and Definition 8 for a proof. We interpret $q^{\text{MSE-AR}}$ as a discrete time version of the Markovian projection [Shi et al., 2024] SDE in Proposition 5 . The reason is because of the second part of Proposition 6 - the model corresponds to a piecewise linear SDE whose form takes almost exactly the same form as the neural SDE. We can see that the only difference between the two SDEs are their Bayes estimators for $\mu^\beta_t(y_{\tau_{1:T}})$:

$$q^{\text{MSE-AR}}: \quad {\mu^\beta_t}^*(x_{t_{1:k}}, y_\mathcal{O}) = \mathbb{E}_{p(y_\mathcal{U}|x_{t_{1:k}}, y_\mathcal{O})}\left[\mu^\beta_t(y_{\tau_{1:T}})\right]$$
$$q^{\text{Neural-SDE}}: \quad {\mu^\beta_t}^*(x_t, x_{t_{1:k}}, y_\mathcal{O}) = \mathbb{E}_{p(y_\mathcal{U}|x_t, x_{t_{1:k}}, y_\mathcal{O})}\left[\mu^\beta_t(y_{\tau_{1:T}})\right]$$

The only difference between the two Bayes estimators is their dependence on the current state x_t. If x_t does not carry more information about $y_\mathcal{U}$ compared to what is already available from $x_{t_{1:k}}$ and $y_\mathcal{O}$, then we can expect that $q^{\text{MSE-AR}}$ and $q^{\text{Neural-SDE}}$ will model nearly the same distribution.

5.5 Discussion

We introduced three different CMFVI based time series models - q^{MSE} 2, $q^{\text{MSE-AR}}$ 6 and $q^{\text{Neural-SDE}}$ 1 which use CMFVI to joint distribution, transition distributions, and infinitesimal transitions of the target distribution respectively. All of these models are Gaussian, and are therefore closely related to existing time series models. First, when one chooses potential functions to align with the data times $\mathcal{R} = \tau_{1:T}$, then q^{MSE} is identical to MSE based non-probabilistic forecasters, which are are trained to predict the future of a time series given an observed history. Next, $q^{\text{MSE-AR}}$ is a conditional Gaussian autoregressive model that is trained to minimize a mean-squared error based objective. This

	Brusselator	Double Pendulum	FitzHugh	Lorenz	Lotka	Van der Pol
MSE	3.04 ± 0.69	9.03 ± 0.34	27.75 ± 4.50	5.91 ± 0.60	2.16 ± 1.18	-0.77 ± 0.01
AR-MSE	0.49 ± 0.18	0.61 ± 0.02	15.08 ± 1.18	8.82 ± 0.29	0.12 ± 0.25	-0.59 ± 0.01
AR-MLE (Latent)	3.39 ± 1.91	0.43 ± 0.01	13.10 ± 2.48	8.49 ± 1.05	0.23 ± 0.27	-0.70 ± 0.00
AR-MLE (Obs.)	3.79 ± 2.05	0.42 ± 0.01	13.35 ± 2.47	7.77 ± 0.76	0.11 ± 0.32	-0.70 ± 0.00
FBGM (Latent)	2.06 ± 1.12	0.56 ± 0.03	6.15 ± 0.75	12.11 ± 0.80	0.17 ± 0.42	-0.69 ± 0.00
FBGM (Obs.)	0.93 ± 0.29	0.51 ± 0.01	11.67 ± 1.80	5.28 ± 0.50	0.47 ± 0.67	-0.71 ± 0.00

(a) Negative log likelihood (lower is better)

	Brusselator	Double Pendulum	FitzHugh	Lorenz	Lotka	Van der Pol
MSE	0.56 ± 0.02	0.99 ± 0.00	2.15 ± 0.16	1.09 ± 0.01	0.50 ± 0.02	0.48 ± 0.00
AR-MSE	0.59 ± 0.01	1.16 ± 0.01	3.58 ± 0.27	1.25 ± 0.01	0.55 ± 0.03	0.52 ± 0.00
AR-MLE (Latent)	0.65 ± 0.04	1.27 ± 0.01	2.32 ± 0.17	1.26 ± 0.03	0.59 ± 0.03	0.52 ± 0.01
AR-MLE (Obs.)	0.66 ± 0.05	1.27 ± 0.01	2.37 ± 0.13	1.26 ± 0.04	0.58 ± 0.03	0.52 ± 0.01
FBGM (Latent)	0.62 ± 0.05	1.20 ± 0.01	2.34 ± 0.17	1.09 ± 0.03	0.55 ± 0.03	0.49 ± 0.01
FBGM (Obs.)	0.64 ± 0.02	1.17 ± 0.01	2.29 ± 0.15	1.08 ± 0.02	0.55 ± 0.03	0.51 ± 0.00

(b) Normalized root mean squared error (lower is better)

Table 1: Evaluation metrics for our models (MSE and AR-MSE) for probabilistic forecasting compared to baseline models trained in both the latent and data spaces.

model is in the same family as conditional Gaussian models that are trained for maximum likelihood, but differ in their parameterization. Finally, $q^{\text{Neural-SDE}}$ is closely related to [Chen et al., 2024a, Park et al., 2024] that construct autoregressive neural SDEs using similar constructions from FBGMs. As discussed in this paper, this class of models is not fundamentally different from simpler, conditional Gaussian autoregressive models and notably offers little from a theoretical perspective over Gaussian chains when the training data is inherently predictable.

6 Experiments

We compare the performance of our models versus other approaches to time series modeling in latent probabilistic forecasting on dynamical system datasets. We created 6 synthetic datasets representing noisy observations of dynamical systems. Our models used a Wiener velocity model as our base SDE and emission potentials of the form $\phi(x_{t_k}|\theta_{t_k}(y_{\tau_{1:N}})) \propto N(y_{t_k}|x_{t_k}, \sigma^2 I)$. Our models, q^{MSE} and $q^{\text{MSE-AR}}$, and the baseline models were trained to approximate the probabilistic forecasting distribution $p(x_{t_{k+1:N}}|x_{t_{1:k}}, y_{\mathcal{O}})$. See Section 15 for details about the datasets, extra experiments, parameters used for stochastic interpolation and other implementation details. Our models, q^{MSE} and $q^{\text{MSE-AR}}$, were each trained using mean squared error to learn their respective Bayes estimators. We used a non-autoregressive FBGM trained with flow-matching and a conditional Gaussian chain trained for maximum likelihood as our baselines. We trained each of these baselines in two ways to learn $p(x_{t_{k+1:N}}|x_{t_{1:k}}, y_{\mathcal{O}})$. First, we trained these baseline models to learn the latent distribution directly by learning directly from samples from $p(x_{t_{1:N}}|y_{\tau_{1:N}})$. Second, we trained these models in the observation space to learn $p(y_{\mathcal{U}}|y_{\mathcal{O}})$ directly, and at test time, produced latent samples $x_{t_{k+1:N}}$ by first sampling $y_{\mathcal{U}}$ using $y_{\mathcal{O}}$, and then sampling from the stochastic interpolator using the full sequence $(y_{\mathcal{O}}, y_{\mathcal{U}})$. For all of the autoregressive models, instead of learning the distribution of the first point $p(x_{t_{k+1}}|y_{\mathcal{O}})$, we produced a heuristic sample by sampling from the stochastic interpolant that is only conditioned on $y_{\mathcal{O}}$. We always chose t_{k+1} to be a time contained in $\mathcal{O}$ in order for this heuristic to give reasonable samples. For each model, we trained using 5 different seeds and report the (empirical) negative log likelihood and normalized root mean squared error of samples from the true distribution, $p(x_{t_{k+1:N}}|y_{\mathcal{U}})$, using 32 sampled trajectories from each model, averaged over each dimension and time step. In all of our models, we used a one layer recurrent neural network with a GRU cell as we found that this model had sufficient model capacity to represent our data. Our results are displayed in Table 1. We can see that there is no model that consistently outperforms any of the others and that the MSE based approaches perform comparably to the maximum likelihood based approaches, which indicates that the choice of maximum likelihood vs CMFVI ultimately depends entirely on the problem setting.

7 Conclusion

We showed how to generalize the recipe underlying flow-based generative models to the time series setting and uncovered a discrete time version of these models that shares convenient properties that FBGMs possess, including a closed form solution and Bayes estimator parameters. This framework unifies a range of existing time series models, including MSE based non-probabilistic forecasters, conditional Gaussian autoregressive models and neural SDEs based on FBGMs. Our insights offer a novel perspective on autoregressive time series models and bridges the gap between modern, flow based generative modeling and traditional time series.

References

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Guan-Horng Liu, Yaron Lipman, Maximilian Nickel, Brian Karrer, Evangelos Theodorou, and Ricky T. Q. Chen. Generalized schrödinger bridge matching. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=SoismgeX7z.

Carles Domingo-Enrich, Michal Drozdzal, Brian Karrer, and Ricky TQ Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. *arXiv preprint arXiv:2409.08861*, 2024.

Aaron Havens, Benjamin Kurt Miller, Bing Yan, Carles Domingo-Enrich, Anuroop Sriram, Brandon Wood, Daniel Levine, Bin Hu, Brandon Amos, Brian Karrer, et al. Adjoint sampling: Highly scalable diffusion samplers via adjoint matching. *arXiv preprint arXiv:2504.11713*, 2025.

Valentin De Bortoli, Guan-Horng Liu, Tianrong Chen, Evangelos A Theodorou, and Weilie Nie. Augmented bridge matching. *arXiv preprint arXiv:2311.06978*, 2023.

Yifan Chen, Mark Goldstein, Mengjian Hua, Michael S. Albergo, Nicholas M. Boffi, and Eric Vanden-Eijnden. Probabilistic forecasting with stochastic interpolants and föllmer processes, 2024a.

Ella Tamir, Najwa Laabid, Markus Heinonen, Vikas Garg, and Arno Solin. Conditional flow matching for time series modelling. In *ICML 2024 Workshop on Structured Probabilistic Inference {\&} Generative Modeling*, 2024.

Byoungwoo Park, Hyungi Lee, and Juho Lee. Efficient modeling of irregular time-series with stochastic optimal control. In *NeurIPS 2024 Workshop on Bayesian Decision-making and Uncertainty*, 2024. URL https://openreview.net/forum?id=KRtuDGFJzu.

Yu Chen, Marin Biloš, Sarthak Mittal, Wei Deng, Kashif Rasul, and Anderson Schneider. Recurrent interpolants for probabilistic time series prediction. *arXiv preprint arXiv:2409.11684*, 2024b.

Yiyuan Yang, Ming Jin, Haomin Wen, Chaoli Zhang, Yuxuan Liang, Lintao Ma, Yi Wang, Chenghao Liu, Bin Yang, Zenglin Xu, et al. A survey on diffusion models for time series and spatio-temporal data. *arXiv preprint arXiv:2404.18886*, 2024.

Caspar Meijer and Lydia Y. Chen. The rise of diffusion models in time-series forecasting, 2024.

Michael Samuel Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2209.15571.

Yuyang Shi, Valentin De Bortoli, Andrew Campbell, and Arnaud Doucet. Diffusion schrödinger bridge matching. *Advances in Neural Information Processing Systems*, 36, 2024.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7): 1661–1674, 2011.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

Edwin T Jaynes. *Probability theory: The logic of science*. Cambridge university press, 2003.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

Calvin Luo. Understanding diffusion models: A unified perspective. *arXiv preprint arXiv:2208.11970*, 2022.

Sander Dieleman. Perspectives on diffusion, 2023. URL https://sander.ai/2023/07/20/perspectives.html.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Simo Särkkä and Arno Solin. *Applied stochastic differential equations*, volume 10. Cambridge University Press, 2019.

Tim Dockhorn, Arash Vahdat, and Karsten Kreis. Score-based generative modeling with critically-damped langevin diffusion. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=CzceR82CYc.

Tianrong Chen, Jiatao Gu, Laurent Dinh, Evangelos Theodorou, Joshua M. Susskind, and Shuangfei Zhai. Generative modeling with phase stochastic bridge. In *The Twelfth International Conference on Learning Representations*, 2024c. URL https://openreview.net/forum?id=tUtGjQEDd4.

Yaakov Bar-Shalom, X. Rong Li, and Thiagalingam Kirubarajan. *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, New York, 2001. ISBN 9780471221272. doi: 10.1002/0471221279. URL https://onlinelibrary.wiley.com/doi/book/10.1002/0471221279.

Simo Särkkä et al. *Recursive Bayesian inference on stochastic differential equations*. Helsinki University of Technology, 2006.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

Marcel Kollovieh, Abdul Fatir Ansari, Michael Bohlke-Schneider, Jasper Zschiegner, Hao Wang, and Yuyang Bernie Wang. Predict, refine, synthesize: Self-guiding diffusion models for probabilistic time series forecasting. *Advances in Neural Information Processing Systems*, 36:28341–28364, 2023.

Xinyu Yuan and Yan Qiao. Diffusion-TS: Interpretable diffusion for general time series generation. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=4h1apFjO99.

Marcel Kollovieh, Marten Lienen, David Lüdke, Leo Schwinn, and Stephan Günnemann. Flow matching with gaussian process priors for probabilistic time series forecasting. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=uxVBbSlKQ4.

Yang Hu, Xiao Wang, Lirong Wu, Huatian Zhang, Stan Z Li, Sheng Wang, and Tianlong Chen. Fm-ts: Flow matching for time series generation. *arXiv preprint arXiv:2411.07506*, 2024.

Kashif Rasul, Calvin Seward, Ingmar Schuster, and Roland Vollgraf. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *International Conference on Machine Learning*, pages 8857–8868. PMLR, 2021.

Macheng Shen and Chen Cheng. Neural sdes as a unified approach to continuous-domain sequence modeling. *arXiv preprint arXiv:2501.18871*, 2025.

Ahmed El-Gazzar and Marcel van Gerven. Probabilistic forecasting via autoregressive flow matching. *arXiv preprint arXiv:2503.10375*, 2025.

John Lafferty, Andrew McCallum, Fernando Pereira, et al. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Icml*, volume 1, page 3. Williamstown, MA, 2001.

Charles Sutton, Andrew McCallum, et al. An introduction to conditional random fields. *Foundations and Trends® in Machine Learning*, 4(4):267–373, 2012.

Raghav Singhal, Mark Goldstein, and Rajesh Ranganath. Where to diffuse, how to diffuse, and how to get back: Automated learning for multivariate diffusions. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=osei3IzUia.

Syeda Sakira Hassan, Simo Särkkä, and Ángel F García-Fernández. Temporal parallelization of inference in hidden markov models. *IEEE Transactions on Signal Processing*, 69:4875–4887, 2021.

Adrien Corenflos, Zheng Zhao, and Simo Särkkä. Gaussian process regression in logarithmic time. *arXiv preprint arXiv*, 2102, 2021.

Jimmy T.H. Smith, Andrew Warrington, and Scott Linderman. Simplified state space layers for sequence modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Ai8Hw3AXqks.

Matthew James Beal. *Variational algorithms for approximate Bayesian inference*. University of London, University College London (United Kingdom), 2003.

Matthew James Johnson et al. *Bayesian time series models and scalable inference*. PhD thesis, Massachusetts Institute of Technology, 2014.

Simo Särkkä and Ángel F García-Fernández. Temporal parallelization of bayesian smoothers. *IEEE Transactions on Automatic Control*, 66(1):299–306, 2020.

Daphane Koller. *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, 2009.

Bernt Øksendal. *Stochastic differential equations*. Springer, 2003.

Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

H. E. Rauch, F. Tung, and C. T. Striebel. Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, 3 (8):1445–1450, 1965.

Emily Beth Fox. *Bayesian nonparametric learning of complex dynamical phenomena*. PhD thesis, Massachusetts Institute of Technology, 2009.

Matthew Johnson and Scott Linderman. pylds: Bayesian inference for linear dynamical systems. https://github.com/mattjj/pylds, 2015. Accessed: 2025-05-07.

8 Appendix

The appendix contains proofs and implementation details for the main paper. It is organized as follows:

1. Background Section 9
   - Exponential family distributions Section 9.1
   - Mean field variational inference Section 9.2
   - Bayes estimation Section 9.3
2. Message passing (10)
   - Sequential message passing (10.1)
   - Parallel message passing (10.2)
   - Basic probabilistic queries (10.4)
3. Conditioned linear SDEs (11)
   - Conditioned linear SDEs (11.1)
   - Basic probabilistic queries (11.2)
   - Corresponding probability flow ODE (11.3)
4. Constrained mean field VI (12)
   - Derivation (12.1)
   - Bayes estimator equivariance (12.2)
   - CMFVI time series models (12.3)
5. Flow-based generative models (13)
   - Score function of FBGMs (13.1)
   - General form of Markovian projection SDE (13.2)
   - General form of Markovian projection ODE (13.3)
6. Message passing implementation details (14)
   - Numerical stability considerations (14.1)
   - Message passing pseudocode (14.2)
7. Dataset details (15)
8. Model implementation details (16)

9 Background

9.1 Exponential family distributions

Our findings can be most easily written using exponential family distributions. Although we restrict our attention to Gaussian distributions, the form of our results are most readable in natural parameter space.

Definition 2 (Exponential family distribution)**.** *An probability distribution is in the exponential family if its density function can be written in the following form:*

$$p(x|\theta) = \exp\{\langle t(x), \theta\rangle - A(\theta)\} \tag{11}$$

where $t(x)$ is called the sufficient statistic, θ the natural parameter and $A(\theta)$ the partition function.

The member of this family that we will use is the multivariate Gaussian distribution. A multivariate Gaussian with mean μ and covariance matrix Σ has the sufficient statistic $t(x) = (x, xx^T)$ and natural parameters $\theta = (-\frac{1}{2}\Sigma^{-1}, \Sigma^{-1}\mu)$. In practice, it is more convenient to drop the $-\frac{1}{2}$ scaling term and work with the parameters $(J, h) = (-\Sigma^{-1}, \Sigma^{-1}\mu)$, where J is the precision matrix of the distribution. While these are not exactly the natural parameters, we will refer to them as so. Throughout this paper, we will work with unnormalized Gaussian distributions, which we call "Gaussian potentials". We use the notation $\phi(x|\theta)$ to denote a Gaussian potential function over x with natural parameters θ. A convenient property of the natural parameter form is that the score function takes a simple form.

$$\nabla \log \phi(x|\theta) = Jx - h \tag{12}$$

Another Gaussian distribution that we will use extensively is the Gaussian transition distribution. We write $\phi_{k+1|k}(x_{k+1}|x_k) = N(x_{k+1}|Ax_k + u, \Sigma)$ to denote the Gaussian transition distribution from x_k to x_{k+1} with state transition matrix A, bias vector u and covariance matrix Σ.

9.2 Mean field variational inference

Mean field variational inference is an approximate inference algorithm for probabilistic models. It's main feature is that it's solution is available in a simple closed form expression. Let $p(x, \theta)$ be a joint distribution over x and θ. The mean field variational problem is to find distributions, $q_x(x)$ and $q_\theta(\theta)$ that minimize the KL divergence between $q_x(x)q_\theta(\theta)$ and $p(x, \theta)$.

Proposition 7 (Mean field variational inference for CRFs)**.** *Let $p(\theta)$ be a distribution over θ, $p(x|\theta)$ be the CRF in Definition 1 and $p(x, \theta) = p(\theta)p(x|\theta)$ be the joint distribution over x and θ. Then the solutions to*

$$\underset{q_x(x), q_\theta(\theta)}{\operatorname{argmin}} \ \mathrm{KL}\left[q_x(x)q_\theta(\theta)|p(x, \theta)\right] \tag{13}$$

will satisfy:

$$q_x(x) \propto \exp\{\mathbb{E}_{q_\theta(\theta)}\left[\log p(x|\theta)\right]\} \tag{14}$$

$$q_\theta(\theta) \propto \exp\{\mathbb{E}_{q_x(x)}\left[\log p(\theta|x)\right]\} \tag{15}$$

See [Beal, 2003] for a proof. Typical use cases of mean field VI use tractable classes of distributions for $p(\theta)$ and $p(x|\theta)$ so that one can perform EM style, alternating updates to obtain the optimal q distributions [Beal, 2003, Johnson et al., 2014]. However, in our setting, we will use mean field VI differently. We will assume nothing about the form of $p(\theta)$, but will constrain the variational problem by fixing $q_\theta(\theta) = p(\theta)$.

9.3 Bayes estimation

Lemma 1 (Bayes estimate of parameter)**.** *Let $p(z, \theta)$ be a joint distribution and let $\theta^*(z)$ be the Bayes estimate of θ based on z under the squared error risk. Then the Bayes estimate takes the following two forms:*

$$\theta^*(z) = \mathbb{E}_{p(\theta|z)}[\theta] = \underset{f(z)}{\operatorname{argmin}} \ \mathbb{E}_{p(z,\theta)}\left[\|f(z) - \theta\|^2\right] \tag{16}$$

Proof. Let $\mathcal{L}[f]$ be the loss function defined as follows:

$$\mathcal{L}[f] = \mathbb{E}_{p(z)}\left[\|f(z) - \theta^*(z)\|^2\right]$$

Clearly, the minimizer of $\mathcal{L}[f]$ is $\theta^*(z)$. With a bit of rearranging and using Bayes rule, we can rewrite $\mathcal{L}[f]$ as follows:

$$\begin{aligned}
\mathcal{L}[f] &= \mathbb{E}_{p(z)}\left[\|f(z) - \theta^*(z)\|^2\right] \\
&= \mathbb{E}_{p(z)}\left[\|f(z)\|^2\right] - 2\mathbb{E}_{p(z)}\left[\langle f(z), \theta^*(z)\rangle\right] + \underbrace{\mathbb{E}_{p(z)}\left[\|\theta^*(z)\|^2\right]}_{\text{const. w.r.t. } f} \\
&= \mathbb{E}_{p(z,\theta)}\left[\|f(z)\|^2\right] - 2\mathbb{E}_{p(z)}\left[\langle f(z), \mathbb{E}_{p(\theta|z)}\left[\theta\right]\rangle\right] + \text{const.} \\
&= \mathbb{E}_{p(z,\theta)}\left[\|f(z)\|^2\right] - 2\mathbb{E}_{p(z,\theta)}\left[\langle f(z), \theta\rangle\right] + \text{const.} \\
&\qquad\qquad \text{(complete the square)} \\
&= \mathbb{E}_{p(z,\theta)}\left[\|f(z) - \theta\|^2\right] - \underbrace{\mathbb{E}_{p(z,\theta)}\left[\|\theta\|^2\right]}_{\text{const. w.r.t. } f} + \text{const.}
\end{aligned}$$

The minimizer of $\mathcal{L}[f]$ is unaffected by the constant terms, and so we have that $\theta^*(z) = \mathbb{E}_{p(\theta|z)}[\theta]$ is the solution to

$$\underset{f(z)}{\operatorname{argmin}} \ \mathbb{E}_{p(z,\theta)}\left[\|\theta - f(z)\|^2\right]$$

□

10 Message passing

In this section we will review message passing and identify the key operations that are needed to perform message passing updates. We defer the discussion of numerically stable implementations of these operations to Section 14. First we'll identify the key operations that are needed to perform message passing updates for the backward messages and then show how these operations can be used to perform message passing updates for the forward messages.

At a high level, the sequential and parallel message passing algorithms are variable elimination algorithms that eliminate different variables of the chain structured graph. The sequential algorithms operates on individual nodes and begins at one of the ends of the chain and sequentially eliminate variable at the end of the chain, whereas the parallel algorithm operates on pairs of nodes and eliminates the middle variable of the pair. For example, a rough sketch of the sequential elimination process looks like $(0), 1, 2, 3, 4 \to (1), 2, 3, 4 \to (2), 3, 4 \to (3), 4 \to (4)$, where the parentheses indicate the current node that is being processed. On the other hand, the parallel algorithm looks like $(0, 1), 2, 3, 4 \to (0, 2), 3, 4 \to (0, 3), 4 \to (0, 4)$.

10.1 Sequential message passing

The sequential message passing updates for the backward messages can be written using the following recurrence relation:

$$\phi(x_{k-1}|\beta_{k-1}) = \int \phi_{k|k-1}(x_k|x_{k-1})\phi(x_k|\theta_k)\phi(x_k|\beta_k)dx_k, \quad \beta_N = 0 \tag{17}$$

See Section 14.3 for pseudocode. There are two operations on Gaussians that are needed to perform these updates. The first is a "multiply" operation that takes two potential functions and returns a new potential function, and the second is an "update" operation that absorbs a potential function into a transition function.

Definition 3 (Multiply)**.** *Let $\phi_1(x)$ and $\phi_2(x)$ be potential functions over the same variable. Then the "multiply" operation is defined as*

$$\phi_1(x)\phi_2(x) \mapsto \hat{\phi}(x) \tag{18}$$

When $\phi_1(x)$ and $\phi_2(x)$ are parameterized using natural parameters, then the multiply operation simply adds the natural parameters, i.e. if θ_1 and θ_2 are the natural parameters of $\phi_1(x)$ and $\phi_2(x)$, then $\phi_1(x|\theta_1)\phi_2(x|\theta_2) \mapsto \phi_1(x|\theta_1 + \theta_2)$.

The second operation is the "update" operation, which absorbs a potential function into a transition function. This operation is what handles the integral in the recurrence relation.

Definition 4 (Update)**.** *Let $\phi(y|x)$ be a transition function and $\phi(y)$ be a potential function over the first variable. Then the "update" operation is defined as*

$$\phi(y)\phi_{y|x}(y|x) \mapsto \hat{\phi}_{y|x}(y|x)\hat{\phi}(x) \tag{19}$$

where $\hat{\phi}_{y|x}(y|x)$ and $\hat{\phi}(x)$ are a new transition function and potential function, respectively.

Essentially, the update operation performs a change of variables of the coupling of x and y on the LHS. Furthermore, when the terms of the LHS are Gaussian, then the terms of the RHS are also Gaussian. This allows us to perform the update operation in closed form (see Section 14).

The multiply and update operations are sufficient to perform the sequential message passing updates for the backward messages. For example, the backward message passing updates can be written as:

$$\int \phi_{k|k-1}(x_k|x_{k-1}) \underbrace{\phi(x_k|\theta_k)\phi(x_k|\beta_k)}_{\text{multiply} \to \phi(x_k|\theta_k+\beta_k)} dx_k \tag{20}$$

$$= \int \underbrace{\phi(x_k|\theta_k + \beta_k)\phi_{k|k-1}(x_k|x_{k-1})}_{\text{update} \to \hat{\phi}_{k|k-1}(x_k|x_{k-1})\phi(x_{k-1}|\beta_{k-1})} dx_k \tag{21}$$

$$= \underbrace{\int \hat{\phi}_{k|k-1}(x_k|x_{k-1})dx_k}_{\text{transition integrates to 1}} \phi(x_{k-1}|\beta_{k-1}) \tag{22}$$

$$= \phi(x_{k-1}|\beta_{k-1}) \tag{23}$$

The forward messages can be computed in a similar manner. The forward messages are given by:

$$\phi(x_{k+1}|\alpha_{k+1}) = \int \phi_{k+1|k}(x_{k+1}|x_k)\phi(x_k|\theta_k)\phi(x_k|\alpha_k)dx_k, \quad \alpha_1 = 0 \tag{24}$$

To find the forward messages, we can exploit the fact that our transition functions are Gaussian and can therefore be reversed. This means that given a transition $\phi(y|x)$, we can find a reversed transition $\phi^T(x|y)$ that evaluates to the same value as $\phi(y|x)$ for all x, y

Definition 5 (Reversed transition)**.** *Let $\phi(y|x)$ be a transition function. Then the reversed transition is defined as*

$$\phi^T(x|y) = \phi(y|x) \tag{25}$$

so that $\phi^T(x|y) = \phi(y|x)$ for all x, y and $\int \phi^T(x|y)dx = \int \phi(y|x)dx = 1$.

Using this reverse operation, we can simply reverse the transition distributions and then find the forward messages by using the same recurrence relation as for the backward messages:

$$\int \underbrace{\phi_{k+1|k}(x_{k+1}|x_k)}_{\text{reverse}} \underbrace{\phi(x_k|\theta_k)\phi(x_k|\alpha_k)}_{\text{multiply} \to \phi(x_k|\theta_k+\alpha_k)} dx_k \tag{26}$$

$$= \int \underbrace{\phi^T(x_k|x_{k+1})\phi(x_k|\theta_k + \alpha_k)}_{\text{update} \to \hat{\phi}^T(x_k|x_{k+1})\phi(x_{k+1}|\alpha_{k+1})} dx_k \tag{27}$$

$$= \underbrace{\int \hat{\phi}^T(x_k|x_{k+1})dx_k}_{\text{transition integrates to 1}} \phi(x_{k+1}|\alpha_{k+1}) \tag{28}$$

$$= \phi(x_{k+1}|\alpha_{k+1}) \tag{29}$$

These message passing updates can be computed in $O(N)$ time using the the multiply, update and reverse operations. However, there is a more efficient way to compute the forward messages using the parallel scan algorithm [Särkkä and García-Fernández, 2020] that reduces the complexity to $O(\log N)$ on parallel compute. We will describe this algorithm in Section 10.2.

10.2 Parallel message passing

In this section we will use slightly different notation to describe the parallel message passing algorithm. We will avoid writing out the parameters of our potential functions and call them by their parameter name. For example, instead of writing $\phi(x_k|\theta_k)$, we will write $\phi_k(x_k)$ and instead of writing $\phi(x_k|\beta_k)$, we will write $\beta(x_k)$.

The building block of the parallel message passing algorithm Särkkä and García-Fernández [2020] is an unnormalized potential function over two variables, which we denote by $\Psi(y, x)$. We assume that $\Psi(y, x)$ can be decomposed into a (normalized) transition distribution and an unnormalized potential function:

$$\Psi(y, x) = \Psi(y|x)\Psi(x) \tag{30}$$

Whenever we write $\Psi(y|x)$, we are referring to a valid conditional probability distribution ($\int \Psi(y|x)dy = 1$). Since $\Psi(y, x)$ is jointly Gaussian over x and y, we are able to integrate out variables in x and y and can also combine neighboring potentials into a new Gaussian potential. These properties allow us to construct a chain operation over potentials that combines neighboring potentials and then integrates out the common variable. We denote this chain operation by $\otimes$:

$$\Psi(y, x) := \int \Psi(y, z)\Psi(z, x)dz =: \Psi(y, z) \otimes \Psi(z, x) \tag{31}$$

An important property of the chain operation is that it is associative due to the fact that we can swap the order or integration (we will prove this in Section 10.3).

A useful perspective of this chain operation is that it amounts to performing variable elimination on the graph defined by the potentials, i.e. performs some sort of message passing [Koller, 2009]. With this in mind, we can perform message passing by constructing the appropriate joint potentials:

Proposition 8 (Parallel messages)**.** *Let $\phi_{k+1|k}$ and ϕ_k be the potential functions for the CRF in Definition 1 and α and β be the messages defined in Eqs.* (17) *and* (24)*. Then*

$$\alpha_k(x_k) = \int \Psi^{fwd}_{1:k}(x_k, x_1)dx_1 \quad \textit{and} \quad \beta_k(x_k) = \int \Psi^{bwd}_{k:N}(x_N|x_k)dx_N \tag{32}$$

where

$$\Psi^{fwd}_{1:k}(x_k, x_1) = \bigotimes_{i=1}^{k-1} \phi_{i+1|i}(x_{i+1}|x_i)\phi_i(x_i) \tag{33}$$

$$\textit{and} \quad \Psi^{bwd}_{k:N}(x_N|x_k) = \bigotimes_{i=N-1}^{k} \phi_{i+1|i}(x_{i+1}|x_i)\phi_{i+1}(x_{i+1}) \tag{34}$$

See appendix Section 10.3 for a proof and Section 16 for pseudocode. Since $\otimes$ is associative, we can evaluate Eq. (33) in $O(\log N)$ time using the parallel scan algorithm [Särkkä and García-Fernández, 2020]. The rough idea is that on parallel compute, one can, in parallel, chain together consecutive pairs of potentials and then recurse on these new chained potentials in order to eventually chain the entire sequence. We provide pseudocode for this a special case of this algorithm in Section 14.3. $\Psi^{\text{fwd}}_{1:k}(x_k, x_1)$ and $\Psi^{\text{bwd}}_{k:N}(x_N|x_k)$ can be thought of as the result of marginalization over the variables between x_1 and x_k and x_k and x_N, respectively.

10.3 Chain operation

Recall that the chain operation is defined in Eq. (31) as

$$\Psi(y, x) := \int \Psi(y, z)\Psi(z, x)dz =: \Psi(y, z) \otimes \Psi(z, x) \tag{35}$$

To see that it is associative, we need to check that $\Psi(y, z) \otimes (\Psi(z, x) \otimes \Psi(x, w)) = (\Psi(y, z) \otimes \Psi(z, x)) \otimes \Psi(x, w)$

$$\Psi(y, z) \otimes (\Psi(z, x) \otimes \Psi(x, w)) = \int \Psi(y, z) \left(\int \Psi(z, x)\Psi(x, w)dx \right) dz \tag{36}$$

$$= \int \int \Psi(y, z)\Psi(z, x)\Psi(x, w)dxdz \tag{37}$$

$$= \int \left(\int \Psi(y, z)\Psi(z, x)dz \right) \Psi(x, w)dx \tag{38}$$

$$= (\Psi(y, z) \otimes \Psi(z, x)) \otimes \Psi(x, w) \tag{39}$$

Proposition 9 (Parallel messages)**.** *Let $\phi_{k+1|k}$ and ϕ_k be the potential functions for the CRF in Definition 1 and α and β be the messages defined in Eqs.* (17) *and* (24)*. Then*

$$\alpha_k(x_k) = \int \Psi^{fwd}_{1:k}(x_k, x_1)dx_1 \quad \textit{and} \quad \beta_k(x_k) = \int \Psi^{bwd}_{k:N}(x_N|x_k)dx_N \tag{40}$$

where

$$\Psi^{fwd}_{1:k}(x_k, x_1) = \bigotimes_{i=1}^{k-1} \phi_{i+1|i}(x_{i+1}|x_i)\phi_i(x_i) \tag{41}$$

$$\textit{and} \quad \Psi^{bwd}_{k:N}(x_N|x_k) = \bigotimes_{i=N-1}^{k} \phi_{i+1|i}(x_{i+1}|x_i)\phi_{i+1}(x_{i+1}) \tag{42}$$

Proof. First for notational clarity, define

$$\Psi^{\text{bwd}}_{i+1,i}(x_{i+1}|x_i) = \phi_{i+1|i}(x_{i+1}|x_i)\phi_{i+1}(x_{i+1}) \quad \text{and} \quad \Psi^{\text{fwd}}_{i+1,i}(x_{i+1}, x_i) = \phi_{i+1|i}(x_{i+1}|x_i)\phi_i(x_i) \tag{43}$$

We can compute the cumulative potentials as follows:

$$\Psi^{\text{bwd}}_{k:N}(x_N|x_k) = \bigotimes_{i=N-1}^{k} \Psi^{\text{bwd}}_{i+1,i}(x_{i+1}|x_i) \tag{44}$$

$$= \Psi^{\text{bwd}}_{N:N-1}(x_N|x_{N-1}) \otimes \Psi^{\text{bwd}}_{N-1:N-2}(x_{N-1}|x_{N-2}) \otimes \cdots \otimes \Psi^{\text{bwd}}_{k+1:k}(x_{k+1}|x_k) \tag{45}$$

$$= \int \Psi^{\text{bwd}}_{N:N-1}(x_N|x_{N-1}) \int \Psi^{\text{bwd}}_{N-1:N-2}(x_{N-1}|x_{N-2})dx_{N-1} \int \Psi^{\text{bwd}}_{N-2:N-3}(x_{N-2}|x_{N-3})dx_{N-2} \cdots dx_{k+1} \tag{46}$$

$$= \int \cdots \int \prod_{i=k}^{N-1} \Psi^{\text{bwd}}_{i:i+1}(x_{i+1}|x_i)dx_{N-1} \cdots dx_{k+1} \tag{47}$$

And similarly for the forward potentials:

$$\Psi^{\text{fwd}}_{1:k}(x_k, x_1) = \bigotimes_{i=1}^{k-1} \Psi^{\text{fwd}}_{i+1,i}(x_{i+1}, x_i) \tag{48}$$

$$= \int \cdots \int \prod_{i=1}^{k-1} \Psi^{\text{fwd}}_{i+1,i}(x_{i+1}, x_i) dx_2 \cdots dx_{k-1} \tag{49}$$

Next, we can rewrite the joint distribution of the CRF in a similar form:

$$p(x_{1:N}) = \prod_{k=1}^{N-1} \phi_{k+1|k}(x_{k+1}|x_k) \prod_{k=1}^{N} \phi_k(x_k) \tag{50}$$

$$= \phi_k(x_k) \prod_{i=k}^{N-1} \Psi^{\text{bwd}}_{i+1,i}(x_{i+1}|x_i) \prod_{i=1}^{k-1} \Psi^{\text{fwd}}_{i+1,i}(x_{i+1}, x_i), \quad \forall k \in \{1, \dots, N\} \tag{51}$$

Then, integrating over the variables $dx_1, \dots, \hat{dx}_k, \dots, dx_N$, where $\hat{dx}_k$ denotes that we are not integrating over x_k, completes the proof:

$$p(x_k) = \int \cdots \int p(x_{1:N}) dx_1 \dots \hat{dx}_k \dots dx_N \tag{52}$$

$$\propto \int \cdots \int \prod_{k=1}^{N-1} \phi_{k+1|k}(x_{k+1}|x_k) \prod_{k=1}^{N} \phi_k(x_k) dx_1 \dots \hat{dx}_k \dots dx_N \tag{53}$$

$$= \phi_k(x_k) \int \cdots \int \prod_{i=k}^{N-1} \Psi^{\text{bwd}}_{i+1,i}(x_{i+1}|x_i) \prod_{i=1}^{k} \Psi^{\text{fwd}}_{i+1,i}(x_{i+1}, x_i) dx_1 \dots \hat{dx}_k \dots dx_N \tag{54}$$

$$= \phi_k(x_k) \underbrace{\int \Psi^{\text{bwd}}_{k:N}(x_N|x_k) dx_N}_{\beta_k(x_k)} \underbrace{\int \Psi^{\text{fwd}}_{1:k}(x_k, x_1) dx_1}_{\alpha_k(x_k)} \tag{55}$$

We can recognize the terms in the last equation as the forward and backward messages, which completes the proof. □

It will be convenient later to define an operator that actually transforms the parameters of the backward messages.

Definition 6 (Backward message passing update operator)**.** *Let $\phi_{k+1|k}(x_{k+1}, x_k)$ be a Gaussian transition function and let $\phi(x_{k+1}|\eta_{k+1})$ be a Gaussian node potential with natural parameters η_{k+1}. Next consider the message passing update:*

$$\phi(x_k|\eta_k) = \int \phi_{k+1|k}(x_{k+1}|x_k) \phi(x_{k+1}|\eta_{k+1}) dx_{k+1} \tag{56}$$

The message passing update operator is denoted by $\Phi_{k,k+1}(\eta_{k+1})$ and is defined to satisfy:

$$\eta_k = \Phi_{k,k+1}(\eta_{k+1}) \tag{57}$$

In particular, the update rule for the backward messages is given by:

$$\beta_k = \Phi_{k,k+1}(\beta_{k+1} + \theta_{k+1}) \tag{58}$$

Corollary 3 (Mixed parameterization backward message update rule)**.** *Let $\phi_{k+1|k}(x_{k+1}|x_k) := N(x_{k+1}|Ax_k + u, \Sigma)$ be a Gaussian transition function and let $\phi(x_{k+1}|\eta_{k+1}) := N(x_{k+1}|\mu, J^{-1})$ be a Gaussian node potential where J is the precision matrix. If η_k and η_{k+1} represent the mean and precision matrix of a Gaussian distribution, then the update and marginalize operator is denoted by $\Phi_{k,k+1}(\eta_{k+1})$ and is given by:*

$$\Phi_{k,k+1}(\mu, J) = \left(A^{-1}(\mu - u), A^T J (I + \Sigma J)^{-1} A\right) \tag{59}$$

Proof. The result follows from Section 14.3. □

10.4 Probabilistic queries

The forward and backward messages can be used to compute the majority of the probabilistic queries of interest on a CRF. Recall our definition of a CRF:

$$p(x_{1:N}|\theta) \propto \prod_{k=1}^{N-1} \phi_{k+1|k}(x_{k+1}|x_k) \prod_{k=1}^{N} \phi(x_k|\theta_k) \tag{60}$$

Next we will describe two probabilistic queries of interest: the marginal distribution and the transition distribution.

Proposition 10 (Marginal distribution)**.**

$$p(x_k|\theta) = \phi(x_k|\theta_k + \alpha_k + \beta_k) \tag{61}$$

Proof. The derivation is given in Eq. (52). For completness, we will change notation:

$$\begin{aligned} p(x_k) &= \phi_k(x_k)\beta_k(x_k)\alpha_k(x_k) \text{ (notation in previous section)} && (62)\\ &:= \phi(x_k|\theta_k)\phi(x_k|\alpha_k)\phi(x_k|\beta_k) \text{ (notation in this section and in main text)} && (63)\\ &= \phi(x_k|\theta_k + \alpha_k + \beta_k) && (64) \end{aligned}$$

□

Proposition 11 (Transition distribution)**.**

$$p(x_{k+1}|x_k, \theta) \propto \phi_{k+1|k}(x_{k+1}|x_k)\phi(x_{k+1}|\theta_{k+1} + \beta_{k+1}) \tag{65}$$

Proof. We can start by computing the joint distribution $p(x_{k+1}, x_k|\theta)$. By using variable elimination, we can show that

$$p(x_{k+1}, x_k|\theta) = \phi(x_k|\alpha_k)\phi_{k+1|k}(x_{k+1}|x_k)\phi(x_{k+1}|\theta_{k+1})\phi(x_{k+1}|\beta_{k+1}) \tag{66}$$

Dividing by the marginal distribution $p(x_k|\theta)$ and using the definition of the transition distribution, we get

$$p(x_{k+1}|x_k, \theta) = \phi_{k+1|k}(x_{k+1}|x_k)\frac{\phi(x_{k+1}|\beta_{k+1} + \theta_{k+1})}{\phi(x_k|\beta_k + \theta_k)} \tag{67}$$

which, after absorbing the denominator into the normalization constant, is equivalent to the desired result. □

Corollary 4 (Autoregressive factorization)**.** *The autoregressive factorization of $p(x_{1:N}|\theta)$ takes the following form:*

$$p(x_{1:N}|\theta) \propto \phi(x_1|\theta_1 + \beta_1)\prod_{k=1}^{N-1}\phi_{k+1|k}(x_{k+1}|x_k)\phi(x_{k+1}|\theta_{k+1} + \beta_{k+1}) \tag{68}$$

Proof. This follows directly from applying Proposition 10 and Proposition 11 to $p(x_{1:N}|\theta) = p(x_1|\theta)\prod_{k=1}^{N-1} p(x_{k+1}|x_k, \theta)$. □

11 Conditioned SDEs

In this section we derive the form of conditioned linear SDEs as well as the corresponding probability flow ODEs.

11.1 Conditioned linear SDE

Proposition 12 (Conditioned Linear SDE)**.** *Let $\phi_{t+s|t}(x_{t+s}|x_t)$ be the transition distribution of the linear SDE $dx_t = (F_t x_t + u_t)dt + L_t dW_t$ and let $\{\phi(x_{t_k}|\theta_{t_k})\}_{t_k\in\mathcal{R}}$ be potential functions at times in the set $\mathcal{R}$. Then the piecewise-linear SDE,*

$$dx_t = (F_t x_t + u_t + L_t L_t^T \nabla \log \phi(x_t|\beta_t))dt + L_t dW_t, \quad x_{t_1} \sim \phi(x_{t_1}|\beta_1 + \theta_1) \tag{69}$$

where $t \in (t_k, t_{k+1})$ and $t_k, t_{k+1} \in \mathcal{R}$, has a joint distribution over any superset of times $t_{1:N} = \mathcal{T} \supseteq \mathcal{R}$ that is given by a CRF:

$$p(x_{t_{1:N}}|\theta) \propto \prod_{t_k\in\mathcal{T}} \phi_{t_{k+1}|t_k}(x_{t_{k+1}}|x_{t_k}) \prod_{t_k\in\mathcal{R}} \phi(x_{t_k}|\theta_{t_k}) \tag{70}$$

where β_t is the extension of the backward message defined in Eq. (17) *to time t:*

$$\phi(x_t|\beta_t) = \int \phi_{t_{k+1}|t}(x_{t_{k+1}}|x_t)\phi(x_{t_{k+1}}|\theta_{t_{k+1}} + \beta_{t_{k+1}})dx_{t_{k+1}} \tag{71}$$

Proof. We will first construct the transition distribution of the conditioned SDE and then use Doob's h-transform to identify the form of the SDE. Recall that Doob's h-transform ([Särkkä and Solin, 2019] section 7.5) is used to find the SDE associated with a transition distribution of the form $p(x_{t+s}|x_t) = \phi_{t+s|t}(x_{t+s}|x_t)\frac{h_{t+s}(x_{t+s})}{h_t(x_t)}$ where $\phi_{t+s|t}(x_{t+s}|x_t)$ is the transition distribution of a base SDE with the form $dx_t = b_t(x_t)dt + L_t dW_t$ and h_t is a

function that satisfies $h_t(x_t) = \int_t^{t+s} \phi_{t+s|t}(x_{t+s}|x_t) h_{t+s}(x_{t+s}) dx_{t+s}$. Then the SDE whose transition distribution is $p(x_{t+s}|x_t)$ is given by

$$dx_t = (b_t(x_t) + L_t L_t^T \nabla \log h_t(x_t))dt + L_t dW_t \tag{72}$$

We will show that the backward messages of the CRF are of the form $h_t(x_t)$ and then use Doob's h-transform to identify the form of the conditioned SDE.

Suppose $t \in (t_k, t_{k+1})$ and $s > 0$ is small enough so that $t+s \in (t_k, t_{k+1})$. Then we can construct the joint distribution over $(t_{t+s}, t_{k+1}, \ldots, t_N)$ given x_t as

$$p(x_{t+s}|x_t) = \int \cdots \int p(x_{t_{k+1:N}}, x_{t+s}|x_t) dx_{t_{k+1}} \cdots dx_{t_N} \tag{73}$$

$$\propto \int \cdots \int \phi(x_{t_{k+1}}|\theta_{t_{k+1}}) \underbrace{\left(\prod_{i=k+1}^{N-1} \phi_{t_{i+1}|t_i}(x_{t_{i+1}}|x_{t_i}) \phi(x_{t_{i+1}}|\theta_{t_{i+1}}) \right)}_{\text{integrate to get parallel bwd message (Proposition 8)}} \phi_{t_{k+1}|t+s}(x_{t_{k+1}}|x_{t+s}) dx_{t_{k+1}} \cdots dx_{t_N} \phi_{t+s|t}(x_{t+s}|x_t) \tag{74}$$

$$= \int \int \phi(x_{t_{k+1}}|\theta_{t_{k+1}}) \Psi^{\text{bwd}}_{k+1:N}(x_{t_N}|x_{t_{k+1}}) \phi_{t_{k+1}|t+s}(x_{t_{k+1}}|x_{t+s}) dx_{t_N} dx_{t_{k+1}} \phi_{t+s|t}(x_{t+s}|x_t) \tag{75}$$

$$= \underbrace{\int \phi(x_{t_{k+1}}|\theta_{t_{k+1}}) \phi(x_{t_{k+1}}|\beta_{t_{k+1}}) \phi_{t_{k+1}|t+s}(x_{t_{k+1}}|x_{t+s}) dx_{t_{k+1}}}_{=:\phi(x_{t+s}|\beta_{t+s})} \phi_{t+s|t}(x_{t+s}|x_t) \tag{76}$$

$$= \phi(x_{t+s}|\beta_{t+s}) \phi_{t+s|t}(x_{t+s}|x_t) \tag{77}$$

We can find the normalizing constant by integrating over x_{t+s}:

$$\int \phi(x_{t+s}|\beta_{t+s}) \phi_{t+s|t}(x_{t+s}|x_t) dx_{t+s} \tag{78}$$

$$= \int \int \phi(x_{t_{k+1}}|\theta_{t_{k+1}}) \phi(x_{t_{k+1}}|\beta_{t_{k+1}}) \phi_{t_{k+1}|t+s}(x_{t_{k+1}}|x_{t+s}) dx_{t_{k+1}} \phi_{t+s|t}(x_{t+s}|x_t) dx_{t+s} \tag{79}$$

$$= \int \phi(x_{t_{k+1}}|\theta_{t_{k+1}}) \phi(x_{t_{k+1}}|\beta_{t_{k+1}}) \underbrace{\int \phi_{t_{k+1}|t+s}(x_{t_{k+1}}|x_{t+s}) \phi_{t+s|t}(x_{t+s}|x_t) dx_{t+s}}_{\phi_{t_{k+1}|t}(x_{t_{k+1}}|x_t)} dx_{t_{k+1}} \tag{80}$$

$$= \int \phi(x_{t_{k+1}}|\theta_{t_{k+1}}) \phi(x_{t_{k+1}}|\beta_{t_{k+1}}) \phi_{t_{k+1}|t}(x_{t_{k+1}}|x_t) dx_{t_{k+1}} \tag{81}$$

$$= \phi(x_t|\beta_t) \tag{82}$$

Therefore, the transition distribution is

$$p(x_{t+s}|x_t) = \phi_{t+s|t}(x_{t+s}|x_t) \frac{\phi(x_{t+s}|\beta_{t+s})}{\phi(x_t|\beta_t)} \tag{83}$$

Note that Eq. (78) also verifies that $\phi(x_t|\beta_t)$ satisfies the normalization condition for $h_t(x_t)$ in Doob's h-transform. Directly applying Doob's h-transform to the transition distribution in Eq. (73) identifies the form of the conditioned SDE:

$$dx_t = (F_t x_t + u_t + L_t L_t^T \nabla \log \phi(x_t|\beta_t))dt + L_t dW_t \tag{84}$$

This piecewise-linear SDE has the correct conditional distribution, $p(x_t|x_{t_{k_1}})$, but requires an initial distribution. One can verify that the initial distribution $p(x_{t_1}) \propto \phi(x_{t_1}|\theta_{t_1} + \beta_{t_1})$ is the first marginal distribution of the CRF in Definition 1. □

11.2 Probabilistic queries for conditioned linear SDEs

Lemma 2 (Marignal distribution of conditioned SDE)**.** *Suppose $t \in (t_k, t_{k+1})$ is a time in between the inducing points t_k and t_{k+1} of the conditioned linear SDE in Proposition 4. Then the marginal distribution of the SDE at time t is given by*

$$p(x_t) = \phi(x_t|\alpha_t + \beta_t) \tag{85}$$

where α_t and β_t are extensions of the forward and backward messages defined in Eq. (24) *and Eq.* (17) *to time t:*

$$\phi(x_t|\alpha_t) = \int \phi_{t|t_{k-1}}(x_t|x_{t_{k-1}})\phi(x_{t_{k-1}}|\theta_{t_{k-1}} + \alpha_{t_{k-1}})dx_{t_{k-1}} \tag{86}$$

and

$$\phi(x_t|\beta_t) = \int \phi_{t|t_{k+1}}(x_t|x_{t_{k+1}})\phi(x_{t_{k+1}}|\theta_{t_{k+1}} + \beta_{t_{k+1}})dx_{t_{k+1}} \tag{87}$$

Proof. We can simply incorporate t into the set discretization times, $t_{1:N}$, used in Proposition 4 to get the desired result. Suppose $t \in (t_i, t_{i+1})$ for some i. Then we can write the joint distribution as

$$p(x_t, x_{t_{1:N}}|\theta) \propto \phi_{t_{i+1}|t_i}(x_{t_{i+1}}|x_{t_i})\phi_{t|t_i}(x_t|x_{t_i}) \prod_{t_k \in \mathcal{T}} \phi_{t_{k+1}|t_k}(x_{t_{k+1}}|x_{t_k}) \prod_{t_k \in \mathcal{R}} \phi(x_{t_k}|\theta_{t_k}) \tag{88}$$

Then we can run variable elimination on the ends of the chain until we are left with the marginal distribution of x_t:

$$\begin{aligned}
p(x_t) &= \int p(x_t, x_{t_{1:N}}|\theta)dx_{t_{1:N}} && (89)\\
&= \int\int \phi(x_{t_i}|\alpha_{t_i} + \theta_{t_i})\phi_{t|t_i}(x_t|x_{t_i})\phi_{t_{i+1}|t}(x_{t_{i+1}}|x_t)\phi(x_{t_{i+1}}|\beta_{t_{i+1}} + \theta_{t_{i+1}})dx_{t_{i+1}}dx_{t_i} && (90)\\
&= \underbrace{\int \phi(x_{t_i}|\alpha_{t_i} + \theta_{t_i})\phi_{t|t_i}(x_t|x_{t_i})dx_{t_i}}_{\phi(x_t|\alpha_t)} \underbrace{\int \phi_{t_{i+1}|t}(x_{t_{i+1}}|x_t)\phi(x_{t_{i+1}}|\beta_{t_{i+1}} + \theta_{t_{i+1}})dx_{t_{i+1}}}_{\phi(x_t|\beta_t)} && (91)\\
&= \phi(x_t|\alpha_t + \beta_t) && (92)
\end{aligned}$$

□

Lemma 3 (Transition distribution of conditioned linear SDE)**.** *Suppose $t \in (t_k, t_{k+1})$ is a time in between the inducing points t_k and t_{k+1} of the conditioned linear SDE in Proposition 4, and suppose that $s > 0$ is small enough so that $t + s \in (t_k, t_{k+1})$. Then the transition distribution of the SDE at time t is given by*

$$\phi_{t+s|t}(x_{t+s}|x_t) \propto \phi_{t+s|t}(x_{t+s}|x_t)\phi(x_{t+s}|\beta_{t+s}) \tag{93}$$

Proof. The proof is embedded in the derivation of the conditioned linear SDE at Eq. (83). □

Corollary 5 (Autoregressive factorization)**.** *The autoregressive factorization of $p(x_{t_{1:N}}|\theta)$ is given by*

$$p(x_{t_{1:N}}|\theta) = p(x_{t_1}|\theta) \prod_{t_k \in \mathcal{T}} \phi_{t_k|t_{k-1}}(x_{t_k}|x_{t_{k-1}})\phi(x_{t_k}|\beta_{t_k}) \tag{94}$$

$$\textit{where } \beta_{t_k} = \begin{cases} \Phi_{t_k,t_{k+1}}(\beta_{t_{k+1}} + \theta_{t_{k+1}}) & \textit{if } t_k \in \mathcal{R} \\ \Phi_{t_k,t_{k+1}}(\beta_{t_{k+1}}) & \textit{otherwise} \end{cases} \tag{95}$$

where $\Phi_{t_k,t_{k+1}}$ is the message passing update operator defined in Definition 6.

Proof. Recall that

$$p(x_{t_{1:N}}|\theta) \propto \prod_{t_k \in \mathcal{T}} \phi_{t_{k+1}|t_k}(x_{t_{k+1}}|x_{t_k}) \prod_{t_k \in \mathcal{R}} \phi(x_{t_k}|\theta_{t_k}) \tag{96}$$

Suppose that for each $t_k \notin \mathcal{R}$, we introduce a new potential function whose natural parameters are 0, which we will denote by $\phi(x_{t_k}|\emptyset_{t_k})$. These new potentials have no effect on the joint distribution, but allow us to rewrite the joint distribution in the same form as in Corollary 4, which yields the result. □

11.3 Probability flow ODE for conditioned linear SDEs

Corollary 6 (Probability flow ODE)**.** *The probability flow ODE of the SDE in Proposition 4 is given by*

$$\frac{dx_t}{dt} = F_t x_t + u_t + \frac{1}{2} L_t L_t^T \left(\nabla \log \phi(x_t|\beta_t) - \nabla \log \phi(x_t|\alpha_t)\right) \tag{97}$$

β_t is the same as in Proposition 4 and α_t is the extension of the forward message defined in Eq. (24) *to time t:*

$$\phi(x_t|\alpha_t) = \int \phi_{t|t_k}(x_t|x_{t_k}) \phi(x_{t_k}|\theta_{t_k} + \alpha_{t_k}) dx_{t_k} \tag{98}$$

Proof. Let $dx_t = b_t(x_t)dt + L_t dW_t$ be an SDE. Then the probability flow ODE is defined Song et al. [2021] as

$$\frac{dx_t}{dt} = b_t(x_t) - \frac{1}{2} L_t L_t^T \nabla \log p_t(x_t) \tag{99}$$

where $p_t(x_t)$ is defined as the marginal distribution of the SDE, which is given by Lemma 2. We can apply this directly to our SDE in Proposition 4 to get the result:

$$\begin{aligned}
\frac{dx_t}{dt} &= (F_t x_t + u_t + L_t L_t^T \nabla \log \phi(x_t|\beta_t)) - \frac{1}{2} L_t L_t^T \nabla \log p_t(x_t) && (100)\\
&= (F_t x_t + u_t + L_t L_t^T \nabla \log \phi(x_t|\beta_t)) - \frac{1}{2} L_t L_t^T (\nabla \log \phi(x_t|\alpha_t) + \nabla \log \phi(x_t|\beta_t)) && (101)\\
&= F_t x_t + u_t + \frac{1}{2} L_t L_t^T \left(\nabla \log \phi(x_t|\beta_t) - \nabla \log \phi(x_t|\alpha_t)\right) && (102)
\end{aligned}$$

□

12 CMFVI proofs

12.1 Constrained mean field VI

Let $\theta \sim p(\theta)$ be an unknown prior distribution on the parameters of the conditional exponential family distribution, $p(x|z,\theta) \propto \exp\{\langle t_z(x), \theta\rangle - A(z,\theta)\}$, where $t_z(x)$ is the sufficient statistic of the exponential family distribution and $A(z,\theta)$ is the log partition function. In our setting, we interpret x and z as unobserved and observed variables and θ as a a parameter that they both depend on. We are interested in performing inference in the predictive distribution $p(x|z)$, where we must integrate out θ. This distribution can be written as:

$$\begin{aligned}
p(x|z) &= \int p(x|z,\theta) p(\theta|z) d\theta && (103)\\
&= \mathbb{E}_{p(\theta|z)} \left[\exp\{\langle t_z(x), \theta\rangle - A(z,\theta)\}\right] && (104)
\end{aligned}$$

where $t_z(x)$ is the sufficient statistic of the conditional exponential family distribution. Since this distribution is intractable, we use a variational approximation to approximate it. Our variational approximation is called the constrained mean field VI approximation and is given by:

$$q^*(x|z) = \underset{q(x|z)}{\operatorname{argmin}} \ \mathrm{KL}\left[q(x|z)p(\theta|z) \| p(x,\theta|z)\right] \tag{105}$$

In this appendix section we will derive facts about $q^*(x|z)$.

Lemma 4 (Alternate constrained mean field VI objectives)**.** *The constrained mean field VI objective,*

$$\mathrm{KL}\left[q(x|z)p(\theta|z) \| p(x,\theta|z)\right] \tag{106}$$

is equal to the following expressions:

1. $$\mathbb{E}_{q(x|z)\, p(\theta|z)} \left[\log \frac{p(\theta|z)}{p(\theta|x,z)}\right] + \mathrm{KL}\left[q(x|z) \| p(x|z)\right] \tag{107}$$

2. $$\mathbb{E}_{q(x|z)\, p(\theta|z)} \left[\log \frac{p(x|z)}{p(x|z,\theta)}\right] + \mathrm{KL}\left[q(x|z) \| p(x|z)\right] \tag{108}$$

3.

$$\mathbb{E}_{q(x|z)}\left[\log q(x|z) - \mathbb{E}_{p(\theta|z)}\left[\log p(x|z,\theta)\right]\right] \tag{109}$$

Proof. The proof is a straightforward rearrangement of terms:

$$\begin{aligned}
\mathrm{KL}\left[q(x|z)p(\theta|z)\|p(x,\theta|z)\right] &= \int\int q(x|z)p(\theta|z)\log\frac{q(x|z)p(\theta|z)}{p(x,\theta|z)}dxdy && (110)\\
&= \int\int q(x|z)p(\theta|z)\log\frac{p(\theta|z)}{p(\theta|x,z)}\frac{q(x|z)}{p(x|z)}dxdy \quad \text{(equals 1)} && (111)\\
&= \int\int q(x|z)p(\theta|z)\log\frac{\cancel{p(x|z)}}{p(x|z,\theta)}\frac{q(x|z)}{\cancel{p(x|z)}}dxdy \quad \text{(equals 2)} && (112)\\
&= \int\int q(x|z)p(\theta|z)\log\frac{q(x|z)}{p(x|z,\theta)}dxdy && (113)\\
&= \mathbb{E}_{q(x|z)}\left[\log q(x|z) - \mathbb{E}_{p(\theta|z)}\left[\log p(x|z,\theta)\right]\right] && (114)
\end{aligned}$$

□

Theorem 2 (Constrained mean field VI solution)**.** *Let $p(x|z,\theta) \propto \exp\{\langle t_z(x),\theta\rangle - A(z,\theta)\}$ be an exponential family distribution and that $\theta \sim p(\theta|z)$. The constrained mean field VI approximation of $p(x|z)$, denoted by $q^*(x|z)$, is defined as follows:*

$$\begin{aligned}
q^*(x|z) &= \operatorname*{argmin}_{q(x|z)} \mathrm{KL}\left[q(x|z)p(\theta|z)\|p(x,\theta|z)\right] && (115)\\
&= p(x|z,\theta^*(z)), \quad \textit{where } \theta^*(z) = \mathbb{E}_{p(\theta|z)}\left[\theta\right] && (116)
\end{aligned}$$

Proof. The proof can follow quickly from the standard mean field VI solutions Beal [2003], but for completeness we will derive it from scratch. Starting from the result of Lemma 4, we have that

$$q^*(x|z) = \operatorname*{argmin}_{q(x|z)} \ \mathbb{E}_{q(x|z)}\left[\log q(x|z) - \mathbb{E}_{p(\theta|z)}\left[\log p(x|z,\theta)\right]\right] \tag{117}$$

We can introduce a Lagrange multiplier to enforce the constraint that the distribution is normalized. Let $q_\epsilon(x|z) = q(x|z) + \epsilon\eta(x|z)$ where η is the variation function and ϵ is a scalar. Then we can take a variation by differentiating with respect to ϵ:

$$\begin{aligned}
&\frac{\partial}{\partial\epsilon}\left(\mathbb{E}_{q_\epsilon(x|z)}\left[\log q_\epsilon(x|z) - \mathbb{E}_{p(\theta|z)}\left[\log p(x|z,\theta)\right]\right] + \lambda\left(\int q_\epsilon(x|z)dx - 1\right)\right) = 0 && (118)\\
\implies &\frac{\partial}{\partial\epsilon}\int q_\epsilon(x|z)\log q_\epsilon(x|z)dx + \int\eta(x|z)\left(\mathbb{E}_{p(\theta|z)}\left[\log p(x|z,\theta)\right] + \lambda\right)dx = 0 && (119)
\end{aligned}$$

The negative entropy term simplies as follows:

$$\begin{aligned}
\frac{\partial}{\partial\epsilon}\int q_\epsilon(x|z)\log q_\epsilon(x|z)dx &= \int\frac{\partial}{\partial\epsilon}q_\epsilon(x|z)\log q_\epsilon(x|z)dx + \int q_\epsilon(x|z)\frac{\partial}{\partial\epsilon}\log q_\epsilon(x|z)dx && (120)\\
&= \int\frac{\partial q_\epsilon(x|z)}{\partial\epsilon}\log q_\epsilon(x|z)dx + \int q_\epsilon(x|z)\frac{\partial\log q_\epsilon(x|z)}{\partial\epsilon}dx && (121)\\
&= \int\eta(x|z)\log q_\epsilon(x|z)dx - \int q_\epsilon(x|z)\frac{1}{q_\epsilon(x|z)}\frac{\partial q_\epsilon(x|z)}{\partial\epsilon}dx && (122)\\
&= \int\eta(x|z)\left(\log q_\epsilon(x|z) - 1\right)dx && (123)
\end{aligned}$$

Plugging this back into the original equation and setting it equal to zero implies that the integrand must be zero:

$$\mathbb{E}_{p(\theta|z)}\left[\log p(x|z,\theta)\right] + \lambda + \log q_\epsilon(x|z) - 1 = 0 \tag{124}$$

Solving for $\log q_\epsilon(x|z)$ (and setting $\epsilon = 0$) yields:

$$\log q(x|z) = \mathbb{E}_{p(\theta|z)}\left[\log p(x|z,\theta)\right] + \lambda - 1 \tag{125}$$

The lagrange multiplier λ ensures that the distribution is normalized, and so we have that

$$q^*(x|z) = \exp\left\{\mathbb{E}_{p(\theta|z)}\left[\log p(x|z,\theta)\right] + \lambda - 1\right\} \tag{126}$$

$$\propto \exp\left\{\mathbb{E}_{p(\theta|z)}\left[\log p(x|z,\theta)\right]\right\} \tag{127}$$

$$\propto \exp\left\{\langle t_z(x), \mathbb{E}_{p(\theta|z)}\left[\theta\right]\rangle\right\} \tag{128}$$

And so we can recognize that $q^*(x|z)$ is in the same exponential family as $p(x|z,\theta)$ but with natural parameter $\mathbb{E}_{p(\theta|z)}\left[\theta\right]$. This completes the proof. □

Next, we emphasize another form of the CMFVI solution that is convenient when deriving CMFVI solutions of other models.

Lemma 5 (Mean field form of CMFVI solution)**.** *The CMFVI approximation of $p(x|z)$ has the following form:*

$$q^*(x|z) \propto \exp\left\{\mathbb{E}_{p(\theta|z)}\left[\log p(x|z,\theta)\right]\right\} \tag{129}$$

Proof. See Eq. (127) □

Corollary 7 (Value of CMFVI objective at optimum)**.** *The value of the CMFVI objective at the optimum is given by:*

$$\mathrm{KL}\left[q^*(x|z)p(\theta|z)\|p(x,\theta|z)\right] = \mathbb{E}_{p(\theta|z)}\left[A(z,\theta)\right] - A(z,\theta^*(z)) \tag{130}$$

where z is fixed, $\theta^*(z) = \mathbb{E}_{p(\theta|z)}\left[\theta\right]$ and $A(z,\theta)$ is the partition function of $p(x|z,\theta)$.

Proof. Let $\theta^*(z) = \mathbb{E}_{p(\theta|z)}\left[\theta\right]$. Recall that $p(x|z,\theta) = \exp\left\{\langle t_z(x),\theta\rangle - A(z,\theta)\right\}$, $q^*(x|z) = p(x|z,\theta^*(z))$ and that the CMFVI objective can be written using an identity from Lemma 4:

$$\mathrm{KL}\left[q(x|z)p(\theta|z)\|p(x,\theta|z)\right] = \mathbb{E}_{q(x|z)}\left[\log q(x|z) - \mathbb{E}_{p(\theta|z)}\left[\log p(x|z,\theta)\right]\right] \tag{131}$$

We can plug $q^*(x|z)$ and $p(x|z,\theta)$ into the identity to get:

$$\mathrm{KL}\left[q^*(x|z)p(\theta|z)\|p(x,\theta|z)\right] \tag{132}$$

$$= \mathbb{E}_{q^*(x|z)}\left[\log q^*(x|z) - \mathbb{E}_{p(\theta|z)}\left[\log p(x|z,\theta)\right]\right] \tag{133}$$

$$= \mathbb{E}_{q^*(x|z)}\left[\left(\langle t_z(x),\theta^*(z)\rangle - A(z,\theta^*(z))\right) - \left(\langle t_z(x), \underbrace{\mathbb{E}_{p(\theta|z)}\left[\theta\right]}_{\theta^*(z)}\rangle - \mathbb{E}_{p(\theta|z)}\left[A(z,\theta)\right]\right)\right] \tag{134}$$

$$= \mathbb{E}_{p(\theta|z)}\left[A(z,\theta)\right] - A(z,\theta^*(z)) \tag{135}$$

□

Proposition 13 (Forward KL divergence)**.** *The forward KL divergence between $p(x|z)$ and $q^*(x|z)$ is given by:*

$$\mathrm{KL}\left[p(x|z)\|q^*(x|z)\right] = -H_p[x|z] - \langle t^*(z),\theta^*(z)\rangle + A(z,\theta^*(z)) \tag{136}$$

where $H_p[x|z]$ is the differential entropy of $p(x|z)$, $t^*(z) = \mathbb{E}_{p(x|z)}\left[t_z(x)\right]$, $\theta^*(z) = \mathbb{E}_{p(\theta|z)}\left[\theta\right]$ and $A(z,\theta)$ is the partition function of $p(x|z,\theta)$.

Proof. This follows from a direct computation:

$$\mathrm{KL}\left[p(x|z)\|q^*(x|z)\right] = -H_p[x|z] - \int p(x|z)\log q^*(x|z)dx \tag{137}$$

$$= -H_p[x|z] - \int p(x|z)\left(\langle t_z(x),\theta^*(z)\rangle - A(z,\theta^*(z))\right)dx \tag{138}$$

$$= -H_p[x|z] - \langle \int p(x|z)t_z(x)dx, \theta^*(z)\rangle + A(z,\theta^*(z)) \tag{139}$$

$$= -H_p[x|z] - \langle t^*(z),\theta^*(z)\rangle + A(z,\theta^*(z)) \tag{140}$$

□

12.2 Bayes estimator equivariance

We will use the equivariance of the Bayes estimator to linear transformations to show that it is also equivariant to message passing updates when the Gaussian potential functions of the corresponding CRF have covariances that only depend on the node index. This result will allow us to reparameterize the Bayes estimator of the backward messages in terms of the previously computed backward messages, and also in terms of the potential function means themselves. This will be useful for relating the CMFVI time series models we construct back traditional time series models, and also for proving that the autoregressive CMFVI model we construct is an approximation of flow-based generative models for time series.

Corollary 8 (Commutativity of Bayes estimator with update and marginalize operator)**.** *Let $\phi_{k+1|k}(x_{k+1}|x_k)$ be a Gaussian transition function and let $\phi(x_{k+1}|\eta_{k+1}) := N(x_{k+1}|\mu_{k+1}(y), J_{k+1}^{-1})$ be a Gaussian node potential where $y \sim p(y)$. Then the Bayes estimator of η_k commutes with the update and marginalize operator. That is,*

$$\mathbb{E}_{p(y)}[\eta_k(y)] = \mathbb{E}_{p(y)}[\Phi_{k,k+1}(\eta_{k+1}(y))] = \Phi_{k,k+1}\left(\mathbb{E}_{p(y)}[\eta_{k+1}(y)]\right) \tag{141}$$

Proof. We can examine the form of $\Phi_{k,k+1}$ from Corollary 3 to see that $\Phi_{k,k+1}$ is linear with respect to $\mu_{k+1}(y)$. Then the result follows from linearity equivariance of the Bayes estimator. □

12.3 CMFVI time series models

Proposition 14 (Naive CMFVI solution)**.** *Let $p(x_{t_{1:N}}|y_{\mathcal{O}})$ be the target distribution. Then the naive CMFVI solution, denoted by $q^{CRF}(x_{t_{1:N}})$ is the CMFVI approximation of $p(x_{t_{1:N}}|y_{\mathcal{O}})$ and is given by:*

$$q^{CRF}(x_{t_{1:N}}) \propto \prod_{t_k \in \mathcal{T}} \phi_{t_{k+1}|t_k}(x_{t_{k+1}}|x_{t_k}) \prod_{t_k \in \mathcal{R}} \phi(x_{t_k}|\theta^*_{t_k}(y_{\mathcal{O}})) \tag{142}$$

where $\theta^*_{t_k}(y_{\mathcal{O}}) = \mathbb{E}_{p(y_{\mathcal{U}}|y_{\mathcal{O}})}[\theta_{t_k}(y_{\tau_{1:T}})]$ is the Bayes estimator of θ_{t_k}.

Proof. By expanding q^* using Lemma 5, one finds that the terms of the log likelihood is linear with respect to $\theta_{t_k}(y_{\tau_{1:T}})$. Then the result follows from the equivariance of the Bayes estimator to linear transformations. □

Proposition 15 (CMFVI transition approximation)**.** *Let $p(x_{t_{1:N}}|y_{\mathcal{O}})$ be the target distribution and consider its k'th autoregressive factor $p(x_{t_k}|x_{t_{1:k-1}}, y_{\mathcal{O}})$. Then the CMFVI transition approximation is given by:*

$$q^{transition}(x_{t_k}|x_{t_{1:k-1}}, y_{\mathcal{O}}) \propto \phi_{t_k|t_{k-1}}(x_{t_k}|x_{t_{k-1}})\phi(x_{t_k}|\beta^*_{t_k}(x_{t_{1:k-1}}, y_{\mathcal{O}})) \tag{143}$$

where $\beta^*_{t_k}(x_{t_{1:k-1}}, y_{\mathcal{O}}) = \mathbb{E}_{p(y_{\mathcal{U}}|x_{t_{1:k-1}}, y_{\mathcal{O}})}[\beta_{t_k}(y_{\tau_{1:T}})]$ is the Bayes estimate of $\beta_{t_k}(y_{\tau_{1:T}})$, which is defined using the message passing update operator $\Phi_{t_k,t_{k+1}}$ from Definition 6 as:

$$\beta_{t_k} = \begin{cases} \Phi_{t_k,t_{k+1}}(\beta_{t_{k+1}}(y_{\tau_{1:T}}) + \theta_{t_{k+1}}(y_{\tau_{1:T}})) & \text{if } t_{k+1} \in \mathcal{R} \\ \Phi_{t_k,t_{k+1}}(\beta_{t_{k+1}}(y_{\tau_{1:T}})) & \text{otherwise} \end{cases} \tag{144}$$

Proof. The transition distribution in the fully observed setting is given by:

$$p(x_{t_k}|x_{t_{1:k-1}}, y_{\tau_{1:T}}) = p(x_{t_k}|x_{t_{k-1}}, y_{\tau_{1:T}}) \tag{145}$$

$$\propto \phi_{t_k|t_{k-1}}(x_{t_k}|x_{t_{k-1}})\phi(x_{t_k}|\beta_{t_k}(y_{\tau_{1:T}})) \tag{146}$$

If we expand the log likelihood of $p(x_{t_k}|x_{t_{1:k-1}}, y_{\tau_{1:T}})$, we would find that the log likelihood is linear with respect to $\beta_{t_k}(y_{\tau_{1:T}})$, and so writing the CMFVI solution using Eq. (127) yields the result. □

We denote this model by $q^{\text{MSE}}(x_{t_{1:N}}|y_{\mathcal{O}})$.

Corollary 9 (MSE Forecaster)**.** *Let $p(x_{t_{1:N}}|y_{\mathcal{O}})$ be the target distribution and suppose the covariances of its potentials are constant with respect to y. Then the MSE-CMFVI solution, denoted by $q^{MSE}(x_{t_{1:N}})$ is the CMFVI approximation of $p(x_{t_{1:N}}|y_{\mathcal{O}})$ obtained by choosing $(x, z, \theta) = (x_{t_{1:N}}, y_{\mathcal{O}}, \theta(y_{\tau_{1:T}}))$:*

$$q^{MSE}(x_{t_{1:N}}|y_{\mathcal{O}}) \propto \prod_{t_k \in \mathcal{T}} \phi_{t_{k+1}|t_k}(x_{t_{k+1}}|x_{t_k}) \prod_{t_k \in \mathcal{R}} N(x_{t_k}|\mu^*_{t_k}(y_{\mathcal{O}}), \Sigma_{t_k}) \tag{147}$$

where $\mu^*_{t_k}(y_{\mathcal{O}}) = \mathbb{E}_{p(y_{\mathcal{U}}|y_{\mathcal{O}})}[\mu_{t_k}(y_{\tau_{1:T}})]$ is the Bayes estimate of μ_{t_k}, and $\phi(x_{t_k}|\theta_{t_k}(y_{\tau_{1:T}})) = N(x_{t_k}|\mu^*_{t_k}(y_{\tau_{1:T}}), \Sigma_{t_k})$.

Proof. This follows from the fact that the potentials are constant with respect to y and the linear equivariance of the Bayes estimator. □

Definition 7 (Autoregressive CMFVI solution)**.** *Let $p(x_{t_{1:N}}|y_{\mathcal{O}})$ be the target distribution. Then the autoregressive CMFVI solution, denoted by $q^{AR}(x_{t_{1:N}})$ is the CMFVI approximation of $p(x_{t_{1:N}}|y_{\mathcal{O}})$ and is given by:*

$$q^{AR}(x_{t_{1:N}}) \propto p(x_{t_1}|y_{\mathcal{O}}) \prod_{t_k \in \mathcal{T}} q^{transition}(x_{t_k}|x_{t_{1:k-1}}, y_{\mathcal{O}}) \tag{148}$$

where $q^{transition}(x_{t_k}|x_{t_{1:k-1}}, y_{\mathcal{O}})$ is the CMFVI transition approximation given by **??**.

Corollary 10 (Autoregressive MSE Forecaster)**.** *Let $p(x_{t_{1:N}}|y_{\mathcal{O}})$ be the target distribution and suppose the covariances of its potentials are constant with respect to y. Then the autoregressive MSE-CMFVI solution, denoted by $q^{AR\text{-}MSE}(x_{t_{1:N}})$ is the CMFVI approximation of $p(x_{t_{1:N}}|y_{\mathcal{O}})$ and is given by:*

$$q^{AR\text{-}MSE}(x_{t_{1:N}}) \propto p(x_{t_1}|y_{\mathcal{O}}) \prod_{t_k \in \mathcal{T}} \phi_{t_k|t_{k-1}}(x_{t_k}|x_{t_{k-1}}) \prod_{t_k \in \mathcal{R}} N(x_{t_k}|\left(\mu_{t_k}^{\beta}\right)^*(x_{t_{1:k}}, y_{\mathcal{O}}), \Sigma_{t_k}^{\beta}) \tag{149}$$

where $\left(\mu_{t_k}^{\beta}\right)^*(x_{t_{1:k}}, y_{\mathcal{O}}) = \mathbb{E}_{p(y_{\mathcal{U}}|x_{t_{1:k}}, y_{\mathcal{O}})}\left[\mu_{t_k}^{\beta}(y_{\tau_{1:T}})\right]$ is the Bayes estimate of $\mu_{t_k}^{\beta}$ and $\Sigma_{t_k}^{\beta}$ is the covariance of the backward message of $p(x_{t_{1:N}}|y_{\tau_{1:T}})$.

Proof. This follows from the fact that the potentials are constant with respect to y and the linear equivariance of the Bayes estimator. □

Definition 8 (Continuous extension of AR-MSE model)**.** *Let q^{AR} be the autoregressive CMFVI solution and consider the setting where the potential functions of $p(x_{t_{1:N}}|y_{\tau_{1:T}})$ have covariances that do not depend on y. Then the continuous extension of q^{AR} is given by the following piecewise linear SDE:*

$$dx_t = (F_t x_t + u_t + L_t L_t^T \nabla \log \phi(x_t|\beta_t^*(x_{t_{1:k}}, y_{\mathcal{O}})))dt + L_t dW_t, \tag{150}$$

$$\text{where } \beta_t^*(x_{t_{1:k}}, y_{\mathcal{O}}) = \mathbb{E}_{p(y_{\mathcal{U}}|x_{t_{1:k}}, y_{\mathcal{O}})}\left[\beta_t(y_{\tau_{1:T}})\right], \quad \text{and } t \in (t_k, t_{k+1}) \tag{151}$$

where $\beta_t^*(x_{t_{1:k}}, y_{\mathcal{O}})$ is the Bayes estimator of $\beta_t(y_{\tau_{1:T}}) = \Phi_{t,t_{k+1}}(\beta_{t_{k+1}}(y_{\tau_{1:T}}))$.

Proof. We just need to verify that this piecewise linear SDE has the same joint distribution as q^{AR} on $t_{1:N}$. To do this, we can just check that each of the linear SDEs that are defined on the intervals (t_k, t_{k+1}) have the same joint distribution as $q^{\text{transition}}(x_{t_k}|x_{t_{1:k-1}}, y_{\mathcal{O}})$ from **??**. □

13 Flow-based generative models proofs

In this section we provide basic results about Bayes estimation for generalized linear stochastic interpolants. Let $dx_t = (F_t x_t + u_t)dt + L_t dW_t$ be the base linear SDE and let the distribution of random draws, at times $t_{1:N}$, be denoted by $p(x_{t_{1:N}}|c)$. Let $p(x_{t_{1:N}}|\theta, c)$ be its conditional distribution given parameters θ that are only available during training time and some extra conditioning information c that is avilable at both training and test time, and suppose that $p(\theta|c)$ is the (unknown) distribution of θ given c. The goal of the techniques in this section (and FBGMs in general), is to construct, and learn, the distribution of $p(x_{t_{1:N}}|c)$, which is the distribution needed to generate samples of $x_{t_{1:N}}$ when we do not have access to the parameters θ. At a high level, FBGMs offer different inference algroithms for this task. In this section, we will derive three of these inference algorithms.

13.1 Score function for FBGMs

Proposition 16 (Score function for FBGMs)**.** *Suppose that $p(\theta|c)$ is a probability distribution over θ given some extra conditioning information c and $p(x_t|\theta, c)$ is the marignal distribution of a generalized linear stochastic interpolant whose base linear SDE is given by $dx_t = (F_t x_t + u_t)dt + L_t dW_t$. Then the score function of $p(x_t|c)$ is given by:*

$$\nabla \log p(x_t|c) = \nabla \log \phi(x_t|\alpha_t^*(x_t, c) + \beta_t^*(x_t, c)) \tag{152}$$

where $\alpha_t^*(x_t, c) = \mathbb{E}_{p(\theta|x_t,c)}\left[\alpha_t(\theta, c)\right]$ and $\beta_t^*(x_t, c) = \mathbb{E}_{p(\theta|x_t,c)}\left[\beta_t(\theta, c)\right]$ are Bayes estimators of the forward and backward messages to time t using x_t respectively.

Proof. A straightforward calculation will lead to the desired result.

$$\nabla \log p(x_t|c) = \frac{1}{p(x_t|c)} \nabla p(x_t|c) \tag{153}$$
$$= \frac{1}{p(x_t|c)} \nabla \int p(\theta|c) p(x_t|\theta, c) d\theta \tag{154}$$
$$= \frac{1}{p(x_t|c)} \int p(\theta|c) \nabla p(x_t|\theta, c) d\theta \tag{155}$$
$$= \int \frac{p(\theta|c)p(x_t|\theta, c)}{p(x_t|c)} \nabla \log p(x_t|\theta, c) d\theta \tag{156}$$
$$= \mathbb{E}_{p(\theta|x_t,c)} \left[\nabla \log p(x_t|\theta, c)\right] \tag{157}$$
$$= \mathbb{E}_{p(\theta|x_t,c)} \left[\nabla \log \phi(x_t|\alpha_t(\theta, c) + \beta_t(\theta, c))\right] \quad \because Lemma\ 2 \tag{158}$$
$$= \nabla \log \phi(x_t|\alpha_t^*(x_t, c) + \beta_t^*(x_t, c)) \quad \because Eq.\ (12) \tag{159}$$

□

13.2 General form of Markovian projection SDE

Lemma 6 (General form of Markovian projection SDE)**.** *Suppose that $p(\theta|c)$ is a probability distribution over θ given some extra conditioning information c and $p(x_t|\theta, c)$ is the marignal distribution of a generalized linear stochastic interpolant whose base linear SDE is given by $dx_t = (F_t x_t + u_t)dt + L_t dW_t$. Then the Markovian projection SDE is given by:*

$$dx_t = (F_t x_t + u_t + L_t L_t^T \nabla \log \phi(x_t|\beta_t^*(x_t, c)))dt + L_t dW_t \tag{160}$$

where $\beta_t^*(x_t, c) = \mathbb{E}_{p(\theta|x_t,c)} \left[\beta_t(\theta, c)\right]$ is the Bayes estimate of the backward message to time t using x_t.

Proof. The Markovian projection SDE is the SDE whose marginal distribution evolves in time in the same way that $p(x_t|c)$ evolves in time, and so our proof strategy will follow the same strategy as [Lipman et al., 2023, Theorem 1] where we take the time derivative of $p(x_t|c)$ and recognize the form of the SDE.

First, recall that the Fokker-Planck equation [Särkkä and Solin, 2019, Øksendal, 2003] relates an SDE to the time derivative of its marginal distribution. Let $p(x_t|\theta, c)$ be the marginal distribution of the generalized linear stochastic interpolant and recall that its corresponding SDE is given by $dx_t = (F_t x_t + u_t + L_t L_t^T \nabla \log \phi(x_t|\beta_t(\theta, c)))dt + L_t dW_t$ (see Proposition 4). Then the Fokker-Planck equation for this SDE is given by:

$$\frac{\partial p(x_t|\theta, c)}{\partial t} = -\text{Div}(p(x_t|\theta, c)(F_t x_t + u_t + L_t L_t^T \nabla \log \phi(x_t|\beta_t(\theta, c)))) + \frac{1}{2} L_t L_t^T \text{Div}(\nabla p(x_t|\theta, c)) \tag{161}$$

$L_t L_t^T$ appears outside the divergence operator because it does not depend on x_t. Next, we can directly take the time derivative of $p(x_t|c)$ and recognize the form of the corresponding SDE.

$$\frac{\partial p(x_t|c)}{\partial t} = \mathbb{E}_{p(\theta|c)} \left[\frac{\partial p(x_t|\theta, c)}{\partial t}\right] \tag{162}$$
$$= \mathbb{E}_{p(\theta|c)} \left[-\text{Div}(p(x_t|\theta, c)(F_t x_t + u_t + L_t L_t^T \nabla \log \phi(x_t|\beta_t(\theta, c)))) + \frac{1}{2} L_t L_t^T \text{Div}(\nabla p(x_t|\theta, c))\right] \tag{163}$$
$$= \mathbb{E}_{p(\theta|c)} \left[-\text{Div}(p(x_t|\theta, c)(F_t x_t + u_t))\right] \quad \text{(A)} \tag{164}$$
$$+ \mathbb{E}_{p(\theta|c)} \left[-\text{Div}(p(x_t|\theta, c) L_t L_t^T \nabla \log \phi(x_t|\beta_t(\theta, c)))\right] \quad \text{(B)} \tag{165}$$
$$+ \mathbb{E}_{p(\theta|c)} \left[\frac{1}{2} L_t L_t^T \text{Div}(\nabla p(x_t|\theta, c))\right] \quad \text{(C)} \tag{166}$$

Since all of the divergence and gradient operators depend only on x_t, we can pass the expectation through these terms. We can simplify each terms as follows:

(A)

$$\mathbb{E}_{p(\theta|c)} \left[-\text{Div}(p(x_t|\theta, c)(F_t x_t + u_t))\right] = -\text{Div}(p(x_t|c)(F_t x_t + u_t)) \tag{167}$$

(B)

$$\mathbb{E}_{p(\theta|c)}\left[-\text{Div}(p(x_t|\theta,c)L_tL_t^T\nabla\log\phi(x_t|\beta_t(\theta,c)))\right] = -\text{Div}(\int p(\theta|c)p(x_t|\theta,c)L_tL_t^T\nabla\log\phi(x_t|\beta_t(\theta,c))d\theta) \tag{168}$$

$$= -\text{Div}(\int p(\theta|x_t,c)p(x_t|c)L_tL_t^T\nabla\log\phi(x_t|\beta_t(\theta,c))d\theta) \tag{169}$$

$$= -\text{Div}(p(x_t|c)L_tL_t^T\mathbb{E}_{p(\theta|x_t,c)}\left[\nabla\log\phi(x_t|\beta_t(\theta,c))\right]) \tag{170}$$

(C)

$$\mathbb{E}_{p(\theta|c)}\left[\frac{1}{2}L_tL_t^T\text{Div}(\nabla p(x_t|\theta,c))\right] = \frac{1}{2}L_tL_t^T\text{Div}(\nabla\mathbb{E}_{p(\theta|c)}\left[p(x_t|\theta,c)\right]) \tag{171}$$

$$= \frac{1}{2}L_tL_t^T\text{Div}(\nabla p(x_t|c)) \tag{172}$$

Putting these terms back together, we get:

$$\frac{\partial p(x_t|c)}{\partial t} = -\text{Div}(p(x_t|c)\underbrace{\left(F_tx_t + u_t + L_tL_t^T\mathbb{E}_{p(\theta|x_t,c)}\left[\nabla\log\phi(x_t|\beta_t(\theta,c))\right]\right)}_{\text{recognize as drift term in Fokker-Planck equation}}) + \frac{1}{2}L_tL_t^T\text{Div}(\nabla p(x_t|c)) \tag{173}$$

We can see that the form of the Markovian projection SDE is given by:

$$dx_t = \left(F_tx_t + u_t + L_tL_t^T\mathbb{E}_{p(\theta|x_t,c)}\left[\nabla\log\phi(x_t|\beta_t(\theta,c))\right]\right)dt + L_tdW_t \tag{174}$$

Lastly because $\phi(x_t|\beta_t(\theta,c))$ is a Gaussian distribution with natural parameters $\beta_t(\theta,c)$, its pdf is given by:

$$\phi(x_t|\beta_t(\theta,c)) = \exp\{\langle t_c(x_t), \beta_t(\theta,c)\rangle - A(c,\theta)\} \tag{175}$$

$$\tag{176}$$

where $t_c(x_t)$ is the sufficient statistic of the Gaussian distribution and $A(c,\theta)$ is the log partition function. From this form, we can immediately see that the expectation around the score function passes through to the natural parameters:

$$\mathbb{E}_{p(\theta|x_t,c)}\left[\nabla\log\phi(x_t|\beta_t(\theta,c))\right] = \langle\nabla t_c(x_t), \mathbb{E}_{p(\theta|x_t,c)}\left[\beta_t(\theta,c)\right]\rangle \tag{177}$$

If we let $\beta_t^*(x_t,c) = \mathbb{E}_{p(\theta|x_t,c)}\left[\beta_t(\theta,c)\right]$ and stop the gradient with respect to x_t through β_t^*, then we recover the desired result. □

Proposition 17 (Neural latent SDE)**.** *Let $p(x_{t_{1:N}}, y_{1:T})$ be the joint distribution defined in* **??** *and suppose that $\mathbf{y} = (y_\mathcal{O}, y_\mathcal{U})$, where $\mathcal{O}$ and $\mathcal{U}$ are the times at which sequences are observed and unobserved, respectively. Then the neural latent SDE is the following piecewise SDE defined on the intervals (t_k, t_{k+1}) for $k = 1,\ldots,N$:*

$$dx_t = (F_tx_t + u_t + L_tL_t^T\nabla\log\phi(x_t|\beta_t^*(x_t, x_{t_{1:k}}, y_\mathcal{O})))dt + L_tdW_t, \tag{178}$$

$$\textit{where } \beta_t^*(x_t, x_{t_{1:k}}, y_\mathcal{O}) = \mathbb{E}_{p(y_\mathcal{U}|x_t,x_{t_{1:k}},y_\mathcal{O})}\left[\beta_t(y_{1:T})\right], \quad \textit{and } t \in (t_k, t_{k+1}) \tag{179}$$

$\beta_t^*(x_t, x_{t_{1:k}}, y_\mathcal{O})$ is the Bayes estimator of β_t using the current state x_t.

Proof. The result follows directly from Lemma 6 by choosing $\theta = y_\mathcal{U}$ and $c = x_{t_{1:k}}$. □

13.3 General form of Markovian projection ODE

Lemma 7 (General form of Markovian projection ODE)**.** *Suppose that $p(\theta|c)$ is a probability distribution over θ given some extra conditioning information c and $p(x_t|\theta,c)$ is the marignal distribution of a generalized linear stochastic interpolant whose base linear SDE is given by $dx_t = (F_tx_t + u_t)dt + L_tdW_t$. Then the Markovian projection ODE is defined as the probability flow ODE of the Markovian projection SDE and is given by:*

$$\frac{dx_t}{dt} = F_tx_t + u_t + \frac{1}{2}L_tL_t^T\left(\nabla\log\phi(x_t|\beta_t^*(x_t,c)) - \nabla\log\phi(x_t|\alpha_t^*(x_t,c))\right) \tag{180}$$

where $\beta_t^*(x_t,c) = \mathbb{E}_{p(\theta|x_t,c)}\left[\beta_t(\theta,c)\right]$ and $\alpha_t^*(x_t,c) = \mathbb{E}_{p(\theta|x_t,c)}\left[\alpha_t(\theta,c)\right]$ are Bayes estimators of the forward and backward messages to time t using x_t respectively.

Proof. Recall that the definition of the probability flow ODE of an SDE of the form $dx_t = b_t(x_t)dt + L_t dW_t$ is given by [Song et al., 2021]:

$$\frac{dx_t}{dt} = b_t(x_t) - \frac{1}{2} L_t L_t^T \nabla \log p(x_t|c) \tag{181}$$

Plugging in drift of the Markovian projection SDE in Lemma 6, and the score function of $p(x_t|c)$ in Proposition 16, we get the desired result. □

14 Message Passing Implementation Details

We devise a careful implementation of message passing to ensure numerical stability. There are many different ways to implement message passing. For example, [Särkkä et al., 2006] parameterizes the potentials in the standard form of Gaussians and uses Kalman filtering [Kalman, 1960] to obtain the forward messages and does not directly compute the backward messages, but instead uses the Rauch-Tung-Striebel smoother [Rauch et al., 1965] to blend the forward and backward message computations to obtain the smoothed potentials. Alternatively, [Fox, 2009, Johnson and Linderman, 2015] utilize a natural parameterization of the potentials in order to have simple message passing updates. Our implementation requires that we can express both total uncertainty, and total certainty, in a variable in order to be able to work with incomplete, or missing data, and to condition exactly on variables. To do this, we adopt a mixed parametrization that contains the mean of the Gaussian and precision matrix so that we can express total uncertainty using a precision matrix of 0 and total certainty in the mean value by using a symbolic infinity. We also use symbolic zeros to mitigate accumulation of errors when perform message passing on long chains of latent variables without any evidence.

14.1 Numerical stability considerations

Before we look at the implementation details, we will look at what considerations we need to make for the implementation of these operations in a numerically stable way. Recall that the transition distribution of an LTI-SDE is given by

$$\phi(x_{t+s}|x_t) = N(x_{t+s}|A_s x_t, \Sigma_s) \tag{182}$$

where

$$\begin{bmatrix} A_s & \Sigma_s A_s^{-T} \\ 0 & A_s^{-T} \end{bmatrix} := \exp\{\begin{bmatrix} F & LL^T \\ 0 & -F^T \end{bmatrix} s\} \tag{183}$$

and that potential functions can be written in natural or standard form as:

$$\phi(x) = \exp\{-\frac{1}{2} x^T J x + x^T h - \log Z\} \tag{184}$$

$$= \exp\{-\frac{1}{2} x^T \Sigma^{-1} x + x^T \Sigma^{-1} \mu - \log Z\} \tag{185}$$

where $\Sigma = J^{-1}$ and $\mu = J^{-1}h$. We assume that the time intervals between consecutive variables are bounded and nonzero so that Σ_s, A_s, and A_s^{-T} are numerically stable. We also assume that the covariance matrices that the user specifies for the node potentials, e.g. Σ or J, are well conditioned. We do not assume that Σ_s^{-1}, Σ^{-1} nor J^{-1} are well conditioned. These assumptions are made to accomodate operations that a user might perform in practice. For example, a user may choose to express 0 certainty in a variable by setting $\Sigma \to \infty$ or $J = 0$ and can choose to express 0 uncertainty by setting $\Sigma = 0$ or $J \to \infty$. Furthermore, if a user chooses to discretize an SDE at points where s is small, or even exactly 0, then Σ_s is close to 0 and so Σ_s^{-1} can be very large. To account for these considerations, we use symbolic computation to represent matrices that are 0 or ∞ as needed. Furthermore, we use three different parameterizations of the Gaussian to ensure that we can handle all cases. We use the **standard** parameterization, (μ, Σ), **natural** parameterization [3], $(J = \Sigma^{-1}, h = \Sigma^{-1}\mu)$, and **mixed** parameterization $(J = \Sigma^{-1}, \mu)$. For brevity, we will not include the updates for the normalizing constant $\log Z$ in our pseudocode.

14.2 Message passing pseudocode

In Section 10 we identified the key operations that are needed to perform variable elimination in the sequential and parallel settings (see Sections 10.1 and 10.2). These operations are:

[3] The true natural parameters are scaled by $-\frac{1}{2}$

1. An "add" operation adds the parameters of two potential functions together (code in Section 14.3).
2. An "update" operation that absorbs a potential function into a transition function (defined in Definition 4 and code in Section 14.3).
3. A "marginalize" operation that marginalizes out a variable from a Gaussian joint distribution. In practice, we fuse this with the "update" operation (code in Section 14.3).
4. A "reverse" operation that reverses the direction of a transition (code in Section 14.3).
5. A "chain" operation that chains two transition functions (defined in Eq. (31) and code in Section 14.3).

In Section 14.3, Section 14.3, Section 14.3, and Section 14.3 we provide pseudocode for message passing that involves these operations.

14.3 Update rules

Now we provide pseudocode for the update rules.

Algorithm 1 `Add`

1. Require: potential functions ϕ_1 and ϕ_2
2. $(J_1, h_1) = \texttt{to\_natural}(\phi_1)$
3. $(J_2, h_2) = \texttt{to\_natural}(\phi_2)$
4. Return $\texttt{from\_natural}((J_1 + J_2, h_1 + h_2))$

Algorithm 2 `Update`

1. Require: potential function ϕ and transition $\phi_{k+1|k}$
2. $(J, \mu) = \texttt{to\_mixed}(\phi)$
3. $(A, u, \Sigma) = \phi_{k+1|k}$
4. $R = J(I + \Sigma J)^{-1}$
5. $S = \Sigma R$
6. $T = I - S$
7. $\bar{\phi}_{k+1|k} = (TA, Tu + S\mu, T\Sigma)$
8. $\bar{\phi} = \texttt{from\_mixed}((A^T R^T A, A^{-1}(\mu - u)))$
9. $\Psi_{k+1,k} = (\bar{\phi}_{k+1|k}, \bar{\phi})$
10. Return $\Psi_{k+1,k}$

Algorithm 3 `Update and marginalize`

1. Require: potential function ϕ and transition $\phi_{k+1|k}$
2. $(\_, \bar{\phi}) = \texttt{Update}(\phi, \phi_{k+1|k})$
3. Return $\bar{\phi}$

Algorithm 4 `Reverse`

1. Require: transition $\phi_{k+1|k}$
2. $(A, u, \Sigma) = \phi_{k+1|k}$
3. $\bar{A} = A^{-1}$
4. $\bar{u} = -A^{-1}u$
5. $\bar{\Sigma} = A^{-1}\Sigma A^{-T}$
6. Return $(\bar{A}, \bar{u}, \bar{\Sigma})$

Algorithm 5 `Chain`

1. Require: transition functions $\phi_{k|k-1}$ and $\phi_{k+1|k}$
2. $A_k, u_k, \Sigma_k = \phi_{k+1|k}$
3. $A_{k-1}, u_{k-1}, \Sigma_{k-1} = \phi_{k|k-1}$
4. $A = A_k A_{k-1}$
5. $u = A_k u_{k-1} + u_k$
6. $\Sigma = \Sigma_k + A_k \Sigma_{k-1} A_k^T$
7. Return (A, u, Σ)

Algorithm 6 `BackwardMessagePassing`

1. Require $(\phi_{2|1}, \ldots, \phi_{N|N-1})$ and $(\phi_1, \ldots, \phi_N)$
2. Initialize $\beta_N = 0$
3. For $k = N, \ldots, 2$:
   (a) $\Psi_{k,k-1} = \texttt{Update}(\phi_{k|k-1}, \phi_k + \beta_k)$
   (b) $\beta_{k-1} = \texttt{Marginalize}(\Psi_{k,k-1})$
4. Return $(\beta_1, \ldots, \beta_N)$

Algorithm 7 `ParallelBackwardMessagePassing`

1. Require $(\phi_{2|1}, \ldots, \phi_{N|N-1})$ and $(\phi_1, \ldots, \phi_N)$
2. In parallel, for $k = N, \ldots, 2$:
   (a) $\Psi_{k,k-1} = \texttt{Update}(\phi_{k|k-1}, \phi_k)$
3. $(\Psi_{1:N}, \ldots, \Psi_{N-1:N}) = \texttt{AssociativeScan}(\texttt{Chain}, \Psi_{2,1}, \ldots, \Psi_{N,N-1})$
4. In parallel, for $k = N-1, \ldots, 1$:
   (a) $\beta_k = \texttt{Marginalize}(\Psi_{k:N})$
5. $\beta_N = 0$
6. Return $(\beta_1, \ldots, \beta_N)$

Algorithm 8 `ForwardMessagePassing`

1. Require $(\phi_{2|1}, \ldots, \phi_{N|N-1})$, $(\phi_1, \ldots, \phi_N)$ and `use_parallel`
2. For $k = 1, \ldots, N-1$:
   (a) $\phi_{k|k+1} = \texttt{Reverse}(\phi_{k+1|k})$
3. If `use_parallel`:
   (a) `MessagePassing` = `ParallelBackwardMessagePassing`
4. Else:
   (a) `MessagePassing` = `BackwardMessagePassing`
5. $(\alpha_N, \ldots, \alpha_1) = \texttt{MessagePassing}((\phi_{N-1|N}, \ldots, \phi_{1|2}), (\phi_N, \ldots, \phi_1))$
6. Return $(\alpha_1, \ldots, \alpha_N)$

Algorithm 9 `AssociativeScan` (Even number of elements only)

1. Require: operator $\oplus$, elements $(t_1, t_2, \ldots, t_n)$ where n is a power of 2
2. If $n == 1$:
   (a) Return t_1
3. In parallel, for $k = 1, \ldots, n/2$:
   (a) $p_k = t_{2k-1} \oplus t_{2k}$
4. $(r_2, r_4, \ldots, r_n) = \texttt{AssociativeScan}(\oplus, (p_1, p_2, \ldots, p_{n/2}))$
5. In parallel, for $k = 1, \ldots, n/2 - 1$:
   (a) $r_{2k+1} = r_{2k} \oplus t_{2k+1}$
6. $r_1 = t_1$
7. Return $(r_1, r_2, \ldots, r_n)$

15 Dataset details

Double pendulum We constructed a synthetic task of forecasting the position and velocity of a double pendulum from noisy observations of the position. We constructed a double pendulum with two rods both with mass and length of 1 and 1 respectively and simulate its motion for $50,000$ seconds and record the Euclidean space coordinates of the endpoints of each rod at a rate of 5Hz. We then add Gaussian noise with a standard deviation of 0.3 to the position data and use the last 10000 points of the data as our full dataset. The conditioned linear SDE that we construct (to model $p(\mathbf{x}|\mathbf{y}_{1:N})$) for this task is a Weiner velocity model with a diffusion coefficient of 4.0 to model the latent space process and Gaussian potential functions centered at the noisy observed positions (with zero padding for the velocity dimensions) and with a standard deviation of 0.3 over the position dimensions and ∞ over the velocity dimensions to ensure that the latent position coordinates can be interpreted as smoothed versions of the observed positions and velocities.

Dynamical systems The remaining datasets are similar to those defined in appendix A.1 of [El-Gazzar and van Gerven, 2025]. The difference in our work is that we do not add process noise to these dynamical systems during simulation and instead add varying amounts of Gaussian noise with a to the trajectories that we generate. For Brusselator, Lotka and Van der Pol we add Gaussian noise with a standard deviation of 0.3 and for FitzHugh and Lorenz we add noise with a standard deviation of 0.2 and 1.0 respectively.

16 Model implementation details

16.1 Base SDE

In all of our experiments, we used the Wiener velocity model [Särkkä and Solin, 2019] as our base SDE. In this model, we assume that the latent variable x_t is the concatenation of a ground truth position, denoted by z_t, and velocity,

denoted by v_t. To capture the physical relationship between position and velocity, we set the relationship $\frac{dz_t}{dt} = v_t$ by constructing F to be the appropriate block matrix. Finally, to ensure that the paths that we sample are smooth, we set L to be 0 in the entries corresponding to z_t. This linear time-invariant SDE takes the following form

$$d\begin{bmatrix} z_t \\ v_t \end{bmatrix} = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} \begin{bmatrix} z_t \\ v_t \end{bmatrix} dt + \begin{bmatrix} 0 & 0 \\ 0 & \sigma I \end{bmatrix} dW_t \tag{186}$$

For the Brusselator, double pendulum, Fitz Hugh, Lorenz, Lotka and Van der Pol datasets, we used a value of σ equal to 0.1, 0.1, 0.379, 0.435, 0.1 and 0.1 respectively.

16.2 Neural network architecture and training details

We used an encoder/decoder neural network architecture to condition on the observed sequence $y_{\tau_{1:T}}$, and generate the latent sequence $x_{t_{1:N}}$. The encoder and decoder for all of our models were a single layer GRU-RNN with a hidden layer size of 128 for all of our models. We initially used a more complex transformer based architecture but found that the simple RNN worked as well in our simple experimental setting. We incorporated information about the times in each series by constructing a feature vector for each scalar time and concatenating it with the observed sequence of variables before passing the concatenation to the RNN.

Each of our models were trained on a single 2080ti GPU using a learning rate of 10^{-4} using the adamw optimizer, linear warmup of 1000 steps, and an effictive batch size of 256 (we used a batch size of 64 and 4 gradient accumulation steps). For each experiment, we used 5 random seeds to initialize the model parameters and to split the data into training, validation, and test sets using an 80/10/10 split. We evaluated the objective function on the entire validation set every 1000 gradient updates and stopped training when the value of the objective function over the entire validation set stopped improving for 5 evaluations.

16.3 Model details

MSE forecaster The MSE forecaster predicts the mean of the potential functions of the CRF used to construct the latent process. This model is trained to minimize the mean squared error between the predicted mean of each potential function, and the mean of the potential function of the target process. To generate samples from this model, we use the input $y_{\tau_{1:k}}$ to generate the means of the CRF potentials for the entire sequence of generated variables. We then sample from the CRF defined by these potentials to get a sample from this model.

Autoregressive (AR) Models The autoregressive models represent a conditional Gaussian chain where the parameters of the next backward message are parametrized by a neural network (see Proposition 6). To generate the next sample of a sequence, an input sequence $y_{\mathcal{O}}$ and history of generated latent values $x_{t_{1:k}}$ are passed to a neural network that outputs the parameters of $\phi(x_{t_{k+1}}|\beta^*_{t_{k+1}}(x_{t_{1:k}}, y_{\mathcal{O}}))$. For the MSE models, we only parameterize the mean of this distribution as in Proposition 6, and for the maximum likelihood models (MLE), we parametrize the mean and diagonal covariance matrix. We then compute the transition distribution $q(x_{t_{k+1}}|x_{t_{1:k}}, y_{\mathcal{O}}) \propto \phi_{t_{k+1}|t_k}(x_{t_{k+1}}|x_{t_k})\beta^*_{t_{k+1}}(x_{t_{1:k}}, y_{\mathcal{O}})$ and then draw the next element of the sequence from this distribution. We compute $q(x_{t_{k+1}}|x_{t_{1:k}}, y_{\mathcal{O}})$ by using the update operation in Section 14.3.

Diffusion model (FBGM) The diffusion model is trained using flow-matching [Lipman et al., 2023] using a brownian bridge between a Gaussian random variable and the sequence of unobserved variables. This model is effectively the same as standard diffusion models for images, but applied to a flattened time series vector. The decoder transformer network outputs the vector field of the probability flow ODE that is used to simulate the process. Samples are generated by passing a sequence of Gaussian random variables of the same size as $y_{\tau_{k+1:N}}$ to an ODE solver that uses the vector field output by the decoder to simulate the process.