Φ-DVAE: LEARNING PHYSICALLY INTERPRETABLE REPRESENTATIONS WITH NONLINEAR FILTERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Incorporating unstructured data into physical models is a challenging problem that is emerging in data assimilation. Traditional approaches focus on well-defined observation operators whose functional forms are typically assumed to be known. This prevents these methods from achieving a consistent model-data synthesis in configurations where the mapping from data-space to model-space is unknown. To address these shortcomings, in this paper we develop a physics-informed dynamical variational autoencoder (Φ-DVAE) for embedding diverse data streams into time-evolving physical systems described by differential equations. Our approach combines a standard (possibly nonlinear) filter for the latent state-space model and a VAE, to embed the unstructured data stream into the latent dynamical system. A variational Bayesian framework is used for the joint estimation of the embedding, latent states, and unknown system parameters. To demonstrate the method, we look at three examples: video datasets generated by the advection and Korteweg-de Vries partial differential equations, and a velocity field generated by the Lorenz-63 system. Comparisons with relevant baselines show that the Φ-DVAE provides a data efficient dynamics encoding methodology that is competitive with standard approaches, with the added benefit of incorporating a physically interpretable latent space.

1 INTRODUCTION

Physical models — as represented by ordinary, stochastic, or partial differential equations — are ubiquitous throughout engineering and the physical sciences. These differential equations are the synthesis of scientific knowledge into mathematical form. However, as a description of reality they are imperfect (Judd & Smith, 2004), leading to the well-known problem of model misspecification (Box, 1979). At least since Kalman (1960) physical modellers have been trying to reconcile their inherently misspecified models with observations (Anderson & Moore, 1979). Such approaches are usually either solving the inverse problem of attempting to recover model parameters from data, and/or, the data assimilation (DA) problem of conducting state inference based on a time-evolving process.

For the inverse problem, Bayesian methods are common (Tarantola, 2005; Stuart, 2010). In this, model parameters $\mathbf{\Lambda}$ are updated with data $\mathbf{y}$ to give a posterior distribution, $p(\mathbf{\Lambda}|\mathbf{y})$. This describes uncertainty with parameters given the data and modelling assumptions. DA can also proceed from a Bayesian viewpoint, where inference is cast as a nonlinear state-space model (SSM) (Law et al., 2015; Reich & Cotter, 2015). The SSM is typically the combination of a time-discretised differential equation and an observation process: uncertainty enters the model through extrusive, additive errors. For a latent state variable $\mathbf{u}_n$ representing some discretised system at time n, with observations $\mathbf{y}_n$, the object of interest is the filtering distribution $p(\mathbf{u}_n|\mathbf{y}_{1:n})$, where $\mathbf{y}_{1:n} := \{\mathbf{y}_k\}_{k=1}^n$. Additionally, the joint filtering and estimation problem, which estimates $p(\mathbf{u}_n, \mathbf{\Lambda}|\mathbf{y}_{1:n})$ has received significant attention in the literature (see, e.g., Kantas et al. (2015) and references therein). This has been well studied in, e.g., electrical engineering (Storvik, 2002), geophysics (Bocquet & Sakov, 2013), neuroscience (Ditlevsen & Samson, 2014), chemical engineering (Kravaris et al., 2013), biochemistry (Dochain, 2003), and hydrology (Moradkhani et al., 2005), to name a few.

In a typical scenario, while parameters of an observation model may be unknown, the observation model itself is typically assumed known (Kantas et al., 2015). This assumption breaks down in settings where data arrives in various modalities, such as videos, images, or audio, hindering the

ability to perform inference. However, in such cases often the underlying variation in the data stream is due to a latent physical process, which is typically at least partially known.

In this work, we address this problem of synthesising known physical models with diverse data streams. The running example of such a data stream, taken in this work, are video data and velocity fields. We develop a variational Bayes (VB) (Blei et al., 2017) methodology which jointly solves the inverse and filtering problems for the case in which the observation operator is unknown. We model this unknown mapping with a variational autoencoder (VAE) (Kingma & Welling, 2014), which encodes the assumed time-dependent observations $\mathbf{y}_{1:N}$ into pseudo-data $\mathbf{x}_{1:N}$ in a latent space. On this latent space, we stipulate that the pseudo-observations are taken from a known dynamical system, given by a stochastic ordinary differential equation (ODE) or partial differential equation (PDE) with possibly unknown coefficients. The differential equation is also assumed to have stochastic forcing, which accounts for possible model misspecification. The stipulated system gives a structured prior $p(\mathbf{x}_{1:N}|\mathbf{\Lambda})$, which acts as a physics-informed regulariser whilst also enabling inference over the unknown $\mathbf{\Lambda}$. This prior is approximated using classical nonlinear filtering algorithms. Our framework is fully probabilistic: inference proceeds from a derived evidence lower bound (ELBO), enabling joint estimation of unknown network parameters and unknown dynamical coefficients via VB. To set the scene for this work, we now review the relevant literature.

2 Related Work

As introduced above, VAEs (Kingma & Welling, 2014) are a popular high-dimensional encoder. A VAE defines a generative model that learns low-dimensional representations, $\mathbf{x}$, of high-dimensional data, $\mathbf{y}$, using VB. To perform efficient inference, a variational approximation $q_\phi(\mathbf{x}|\mathbf{y})$ is made to the intractable posterior $p(\mathbf{x}|\mathbf{y})$. Variational parameters ϕ are estimated via optimisation of the derived ELBO. This unsupervised learning approach infers latent representations of high-dimensional data. Recent works have extended the VAE to high-dimensional time-series data $\mathbf{y}_{1:N}$, indexed by time n, with the aim of jointly learning latent representations $\mathbf{x}_{1:N}$, and a dynamical system that evolves them. These dynamical variational autoencoder (DVAE) methods (Girin et al., 2021) enforce the dynamics with a structured prior $p(\mathbf{x}_{1:N})$ on the latent space.

Various DVAE methods have been proposed. The Kalman variational autoencoder (KVAE) Fraccaro et al. (2017) is a popular approach, which encodes $\mathbf{y}_{1:N}$ into latent variables $\mathbf{x}_{1:N}$ that are assumed to be observations of a linear Gaussian state-space model (LGSSM), driven by latent dynamic states $\mathbf{u}_{1:N}$. Assumed linear dynamics are jointly learnt with the encoder and decoder, via Kalman filtering/smoothing. Another approach is the Gaussian process variational autoencoder (GPVAE) (Pearce, 2020; Jazbec et al., 2021; Fortuin et al., 2020), which models $\mathbf{x}_{1:N}$ as a temporally correlated Gaussian process (GP). The Markovian variant of Zhu et al. (2022) allows for a similar Kalman procedure as in the KVAE, except, in this instance, the dynamics are known and are given by an stochastic differential equation (SDE) approximation to the GP (Hartikainen & Sarkka, 2010). A related approach is provided for control applications in Watter et al. (2015); Hafner et al. (2019), where locally linear embeddings are estimated. Yildiz et al. (2019) also propose the so-called ODE^{2}VAE, which encodes the data to an initial condition which is integrated through time using a Bayesian neural ODE (Chen et al., 2018). This trajectory, only, is used to generate the reconstructions via the decoder network.

Another related class of methods are called deep SSMs, see, e.g., Bayer & Osendorfer (2014), Krishnan et al. (2015), and Karl et al. (2017). These works assume that the parametric form of the SSM is unknown, and replace these transition and emission distributions with neural network models, which are trained based on an ELBO. They harness the representational power of deep neural networks to directly model transitions between high-dimensional states. More emphasis is placed on generative modelling and prediction than representation learning, or system identification. Similarly in Chung et al. (2015) the variational recurrent neural networks (VRNN) attempt to capture variation in highly structured time-series data, by pairing a recurrent neural network for learning nonlinear state-transitions with a sequential latent random variable model.

Methods to include physical information inside of autoencoders have been studied in the physics community. A popular approach uses SINDy (Brunton et al., 2016) for discovery of low-dimensional latent dynamical systems using autoencoders (Champion et al., 2019). A predictive framework is given in Lopez & Atzberger (2021), which aims to learn nonlinear dynamics by jointly optimizing

an ELBO. Following our notation, this learns some function which maps $\mathbf{u}_n \mapsto \mathbf{u}_{n+k}$, for some k, via a VAE. Lusch et al. (2018) use a physics-informed autoencoder to linearise nonlinear dynamical systems via a Koopman approach; inference is regularised through incorporating the Koopman structure in the loss function. Otto & Rowley (2019) present a similar method, and an extension of these approaches to PDE systems is given in Gin et al. (2021). Morton et al. (2018) use the linear regression methods of Takeishi et al. (2017) within a standard autoencoder to similarly compute the Koopman observables. Erichson et al. (2019) derive an autoencoder which incorporates a linear Markovian prediction operator, similar to a Koopman operator, which uses physics-informed regulariser to promote Lyapunov stability. We note also the related works of Hernández et al. (2018) and Lu et al. (2020) which look at VAE methods to encode high-dimensional dynamical systems and dynamic parameters, respectively.

Our contribution. In this paper we propose a physics-informed dynamical variational autoencoder (Φ-DVAE): a DVAE approach which imposes the additional structure of known physics on the latent space. We assume that there is a low-dimensional dynamical system generating the high-dimensional observed time-series: neural networks are used to learn the unknown embedding in this lower dimensional space. On the lower-dimensional space, the embedded data are pseudo-observations of a latent dynamical system, which is, in general, derived from a numerical discretisation of a nonlinear PDE. However, the methodology is suitably generic, allowing for ODE latent systems, and other PDE discretisations. Inference on this latent system is done with efficient nonlinear stochastic filtering methods, enabling the use of mature DA algorithms within our framework. Our approach follows a probabilistically coherent VB construction and allows for joint learning of both the embedding and unknown dynamical parameters. Our Φ-DVAE trades-off between the ability to discover the latent system and infer physical parameters and states, solving the joint filtering and parameter estimation problem in scenarios where the observation model is unknown.

3 The Probabilistic Model

In this section we define our probabilistic model; our presentation roughly follows the structure of the generative model. We first give an overview of the dependencies between variables, as described by conditional probabilities. We then cover the latent differential equation model used to describe the underlying physics. Then, the pseudo-observation model is covered, followed by the decoder and the encoder. To be precise, we assume a general SSM:

$$\mathbf{\Lambda} \sim p(\mathbf{\Lambda}), \tag{1}$$

$$\mathbf{u}_n|\mathbf{u}_{n-1}, \mathbf{\Lambda} \sim p(\mathbf{u}_n|\mathbf{u}_{n-1}, \mathbf{\Lambda}), \tag{2}$$

$$\mathbf{x}_n|\mathbf{u}_n \sim p_\nu(\mathbf{x}_n|\mathbf{u}_n), \tag{3}$$

$$\mathbf{y}_n|\mathbf{x}_n \sim p_\theta(\mathbf{y}_n|\mathbf{x}_n), \tag{4}$$

where $\mathbf{\Lambda}$ describes the parameters of the Markov process $\{\mathbf{u}_n\}_{n=0}^N$ evolving w.r.t. the *dynamic model* $p(\mathbf{u}_n|\mathbf{u}_{n-1}, \mathbf{\Lambda})$, ν describes the parameters of the *likelihood* denoted $p_\nu(\mathbf{x}_n|\mathbf{u}_n)$, and θ describes neural network parameters for the *decoder* $p_\theta(\mathbf{y}_n|\mathbf{x}_n)$. Intuitively, $\{\mathbf{y}_n\}_{n=1}^N$ describes the sequence of high-dimensional video frames (e.g. the footage of a physical phenomenon), while $\{\mathbf{x}_n\}_{n=1}^N$ is its embedding (also referred to as pseudo-data), and $\{\mathbf{u}_n\}_{n=0}^N$ is the *latent physics process*. For each n, we assume that $\mathbf{y}_n \in \mathcal{Y}$ (with $\dim(\mathcal{Y}) = n_y$), $\mathbf{x}_n \in \mathbb{R}^{n_x}$, $\mathbf{u}_n \in \mathbb{R}^{n_u}$, and $\mathbf{\Lambda} \in \mathbb{R}^{n_\lambda}$. In what follows, we describe the components of our probabilistic model in detail.

3.1 Dynamic Model

The first component of the generative model is the latent dynamical system $p(\mathbf{u}_n|\mathbf{u}_{n-1}, \mathbf{\Lambda})$. In general, we model this latent physics process $\{\mathbf{u}_n\}_{n=0}^N$ as a discretised stochastic PDE, however ODE latent physics is admissible within our framework (see Section 5.1). We discretise this process with the statistical finite element method (statFEM) (Girolami et al., 2021; Duffin et al., 2021; 2022; Akyildiz et al., 2022), forming the basis of the physics-informed prior on the latent space. Stochastic additive forcing inside the PDE represents additive model error, which results from possibly misspecified physics. Full details, including the ODE case, are given in Appendix C.

In general, we assume that the model has possibly unknown coefficients $\mathbf{\Lambda}$; on these we place the Bayesian prior $\mathbf{\Lambda} \sim p(\mathbf{\Lambda})$ (Stuart, 2010), describing our *a priori* knowledge on the model parameters

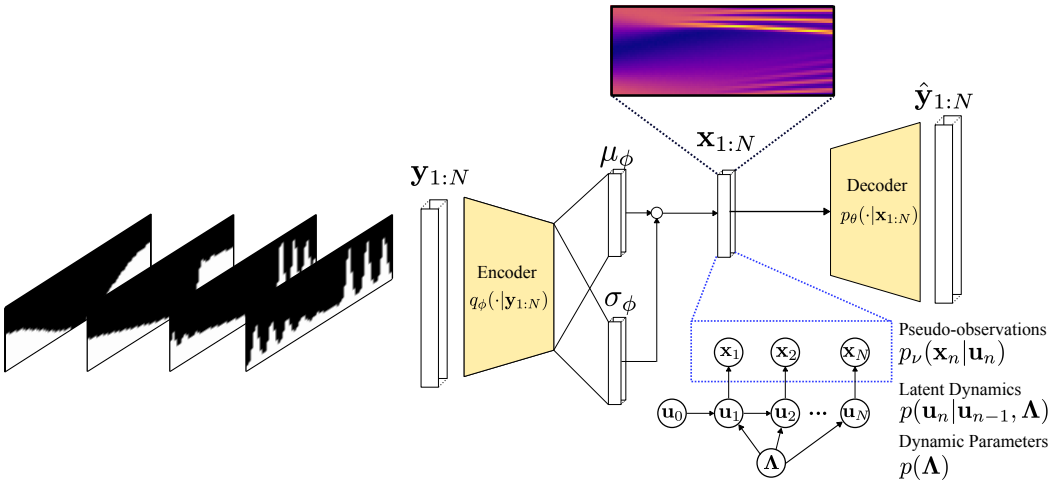


Figure 1: An illustration of the Φ-DVAE model. On the left, the frames of a video can be seen which are denoted $\mathbf{y}_{1:N}$. These are converted into physically interpretable low-dimensional encodings $\mathbf{x}_{1:N}$ using an encoder. This learning process is informed by the physics driven state-space model which treats $\mathbf{x}_{1:N}$ as *pseudo-observations*, which can be seen on the bottom right.

before observing any data. We also assume that $\mathbf{u}_0$ is known up to measurement noise, with prior $p(\mathbf{u}_0)$ set accordingly. To obtain the discrete-time dynamic model we start from a generic nonlinear PDE system of the form

$$\partial_t u + L_\Lambda u + F_\Lambda(u) = f + \dot{\xi}, \quad \dot{\xi} \sim \mathcal{GP}(0, \delta(t-t') \cdot k(\mathbf{s}, \mathbf{s}')), \tag{5}$$

where $u := u(\mathbf{s}, t) \in \mathbb{R}$, $\xi := \xi(\mathbf{s}, t)$, $f := f(\mathbf{s})$, $\mathbf{s} \in \Omega \subset \mathbb{R}^d$, and $t \in [0, T]$. Informally $\dot{\xi}$ is the derivative of a function-valued Wiener process (Da Prato & Zabczyk, 2014), whose increments are given by Gaussian process with the covariance kernel $k(\cdot,\cdot) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ (Williams & Rasmussen, 2006). From a Bayesian perspective, this is an uncertain term in the PDE, representing possible model misspecification with spatial correlation. The operators L_Λ and $F_\Lambda(\cdot)$ are linear and nonlinear differential operators, respectively, with parameters $\mathbf{\Lambda}$. Note that we deal with scalar fields $u(\mathbf{s}, t)$ only however the approach similarly holds for vector fields.

Following STATFEM, the equations are discretised spatially with finite elements first, then discretised in time; this follows a standard method of lines approach (Schiesser, 1991). Thus, equation 5 is multiplied with a sufficiently smooth test function $v \in V$, where V is an appropriate function space, and integrated over the domain Ω to give the weak form (Brenner & Scott, 2008; Thomée, 2006)

$$\langle \partial_t u, v \rangle + \mathcal{A}_\Lambda(u, v) + \langle F_\Lambda(u), v \rangle = \langle f, v \rangle + \langle \dot{\xi}, v \rangle.$$

Note $\mathcal{A}_\Lambda(\cdot,\cdot)$ is the bilinear form generated from the linear operator L_Λ, and $\langle \cdot, \cdot \rangle$ is the $L^2(\Omega)$ inner product.

The domain is discretised to give the mesh $\Omega_h \subseteq \Omega$ with vertices $\{\mathbf{s}_j\}_{j=1}^{n_h}$. On the mesh a finite-dimensional set of polynomial basis functions $\{\phi_j(\mathbf{s})\}_{j=1}^{n_u}$ is defined, such that $\phi_i(\mathbf{s}_j) = \delta_{ij}$. Letting $u_h(\mathbf{s}, t) = \sum_{j=1}^{n_u} u_j(t)\phi_j(\mathbf{s})$, the weak form is now rewritten with these basis functions

$$\langle \partial_t u_h, \phi_j \rangle + \mathcal{A}_\Lambda(u_h, \phi_j) + \langle F_\Lambda(u_h), \phi_j \rangle = \langle f, \phi_j \rangle + \langle \dot{\xi}, \phi_j \rangle, \quad j = 1, \ldots, n_u.$$

This gives a finite-dimensional SDE over the FEM coefficients $\mathbf{u}(t) = (u_1(t), \ldots, u_{n_u}(t))^\top$:

$$\mathbf{M}\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} + \mathbf{A}\mathbf{u} + \mathcal{F}(\mathbf{u}) = \mathbf{b} + \dot{\boldsymbol{\xi}}, \quad \dot{\boldsymbol{\xi}}(t) \sim \mathcal{N}(\mathbf{0}, \delta(t-t') \cdot \mathbf{G}),$$

where $\mathbf{M}_{ij} = \langle \phi_i, \phi_j \rangle$, $\mathbf{A}_{ij} = \mathcal{A}_\Lambda(\phi_i, \phi_j)$, $\mathcal{F}(\mathbf{u})_j = \langle F_\Lambda(u_h), \phi_j \rangle$, $\mathbf{b}_j = \langle f, \phi_j \rangle$, and $\mathbf{G}_{ij} = \langle \phi_i, \langle k(\cdot,\cdot), \phi_j \rangle \rangle$.

An Euler-Maruyama time discretisation (Kloeden & Platen, 1992) eventually gives a transition model

$$p(\mathbf{u}_n | \mathbf{u}_{n-1}, \Lambda) = \mathcal{N}\left(\mathcal{M}(\mathbf{u}^{n-1}), \mathbf{Q}\right), \tag{6}$$

$$\mathcal{M}(\mathbf{u}^{n-1}) = (\mathbf{I} - \Delta_t \mathbf{M}^{-1}\mathbf{A})\mathbf{u}_{n-1} - \Delta_t \mathbf{M}^{-1}\mathcal{F}(\mathbf{u}^{n-1}) + \Delta_t \mathbf{M}^{-1}\mathbf{b}, \quad \mathbf{Q} = \Delta_t \mathbf{M}^{-1}\mathbf{G}\mathbf{M}^{-\top}.$$

The general form of the transition density is discretisation-dependent; see the Appendix C for full details.

3.2 LIKELIHOOD

The second component of the generative model is the *likelihood* $p_\nu(\mathbf{x}_n|\mathbf{u}_n)$. This density acts as a *data likelihood* for embedding $\{\mathbf{x}_n\}_{n=1}^N$ (pseudo-data). We note that this *middle layer* in the model is usually necessary, as the high-dimensional observations $\{\mathbf{y}_n\}_{n=1}^N$ might be generated only some *observed* dimensions of $\{\mathbf{u}_n\}_{n=0}^N$, therefore, this explicit likelihood is introduced to obtain a well-defined formulation for embedding $\{\mathbf{x}_n\}_{n=1}^N$ and to separate the encoding process from the filtering problem (Fraccaro et al., 2017).

The latent states $\mathbf{u}_n$ are mapped at discrete times to "pseudo-observations" via

$$\mathbf{x}_n = \mathbf{H}\mathbf{u}_n + \mathbf{r}_n, \quad \mathbf{r}_n \sim \mathcal{N}(\mathbf{0}, \mathbf{R}). \tag{7}$$

We parameterise this density as $p_\nu(\mathbf{x}_n|\mathbf{u}_n)$ where $\nu = \{\mathbf{H}, \mathbf{R}\}$. Both the pseudo-observation operator $\mathbf{H} \in \mathbb{R}^{n_x \times n_u}$ and the noise covariance $\mathbf{R} \in \mathbb{R}^{n_x \times n_x}$ are assumed to be known in this work. An additional noise process is assumed, $\mathbf{r}_n$, to represent extraneous uncertainty associated with the pseudo-observations. Observations $\mathbf{y}_n$ are related to pseudo-observations $\mathbf{x}_n$ via the decoder, represented with the conditional density $p_\theta(\mathbf{y}_n|\mathbf{x}_n)$. This is discussed in more detail in the subsequent section.

The combination of the transition and the observation densities (equation 6 and equation 7, respectively) provides the nonlinear Gaussian SSM:

$$\begin{aligned} \text{Transition:} \quad & \mathbf{u}_n = \mathcal{M}(\mathbf{u}_{n-1}) + \mathbf{e}_{n-1}, \quad \mathbf{e}_{n-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}), \\ \text{Observation:} \quad & \mathbf{x}_n = \mathbf{H}\mathbf{u}_n + \mathbf{r}_n, \quad \mathbf{r}_n \sim \mathcal{N}(\mathbf{0}, \mathbf{R}). \end{aligned}$$

3.3 DECODER

The last component of our generative model is the decoder $p_\theta(\mathbf{y}_n|\mathbf{x}_n)$, a probabilistic object which describes the unknown mapping between the latent dynamical system pseudo-observations $\mathbf{x}_n$, and the observed data $\mathbf{y}_n$. The decoding of latents to data should model as closely as possible the true data generation process, and prior knowledge about this process can be used to select an appropriate $p_\theta(\mathbf{y}_n|\mathbf{x}_n)$. The reconstruction is given by a differentiable parameterised mapping $\mu_\theta : \mathbb{R}^{n_x} \to \mathcal{Y}$. For $\mathcal{Y} = \mathbb{R}^{n_y}$, we specify $p_\theta(\mathbf{y}_n|\mathbf{x}_n) = \mathcal{N}(\mu_\theta(\mathbf{x}_n), \eta^2\mathbf{I})$. With gray-scale image data, i.e. $\mathcal{Y} = [0,1]^{n_y}$, the decoder maps to a Bernoulli probability vector $p_\theta(\mathbf{y}_n|\mathbf{x}_n) = \text{Bern}(\mu_\theta(\mathbf{x}_n))$. We assume that there is no temporal structure on θ: the decoder is shared across all times: $p_\theta(\mathbf{y}_{1:N}|\mathbf{x}_{1:N}) = \prod_{n=1}^N p_\theta(\mathbf{y}_n|\mathbf{x}_n)$. Unless otherwise specified this parameterised mapping is given by a neural network. For more details about the specific architectures we use, see Appendix A.

4 VARIATIONAL INFERENCE

In this section, we introduce the variational family and derive the ELBO. Denote by $p(\mathbf{y}_{1:N}, \mathbf{x}_{1:N}, \mathbf{u}_{1:N}, \mathbf{\Lambda})$ the joint distribution of the data with all the latent variables, including physical parameters, and denote by $q(\mathbf{u}_{1:N}, \mathbf{x}_{1:N}, \mathbf{\Lambda}|\mathbf{y}_{1:N})$ the variational posterior. The conditional independence structure imposed by the model gives

$$p(\mathbf{y}_{1:N}, \mathbf{x}_{1:N}, \mathbf{u}_{1:N}, \mathbf{\Lambda}) = p_\theta(\mathbf{y}_{1:N}|\mathbf{x}_{1:N}) p(\mathbf{x}_{1:N}|\mathbf{u}_{1:N}) p(\mathbf{u}_{1:N}|\mathbf{\Lambda}) p(\mathbf{\Lambda}). \tag{8}$$

Next, we introduce the variational family $q(\cdot)$, similar to Fraccaro et al. (2017), which factorises as

$$q(\mathbf{u}_{1:N}, \mathbf{x}_{1:N}, \mathbf{\Lambda}|\mathbf{y}_{1:N}) = p(\mathbf{u}_{1:N}|\mathbf{x}_{1:N}, \mathbf{\Lambda}) q_\phi(\mathbf{x}_{1:N}|\mathbf{y}_{1:N}) q_\lambda(\mathbf{\Lambda}). \tag{9}$$

Given the model and the variational family, we derive the ELBO (see also Appendix B):

$$\mathcal{F}(\theta, \phi, \lambda) = \mathbb{E}_{q_\phi}\left[\log \frac{p_\theta(\mathbf{y}_{1:N}|\mathbf{x}_{1:N})}{q_\phi(\mathbf{x}_{1:N}|\mathbf{y}_{1:N})} + \mathbb{E}_{q_\lambda}\left[\log p(\mathbf{x}_{1:N}|\mathbf{\Lambda}) + \log \frac{p(\mathbf{\Lambda})}{q_\lambda(\mathbf{\Lambda})}\right]\right]. \tag{10}$$

Typically this expectation is not analytically tractable and Monte Carlo is used to compute an approximation.

Nonlinear Filtering. In the ELBO of equation 10, the critical term to estimate is $\log p(\mathbf{x}_{1:N}|\mathbf{\Lambda})$. This requires marginalising over $\mathbf{u}_{1:N}$, *the latent physics process*, which is equivalent to computing the marginal likelihood in a nonlinear filtering context. We perform this via the extended Kalman filter (ExKF) (see, e.g., Jazwinski, 1970; Law et al., 2015), which recursively computes a Gaussian approximation to the filtering posterior $p(\mathbf{u}_n|\mathbf{x}_{1:n}, \mathbf{\Lambda}) \approx \mathcal{N}(\mathbf{m}_n, \mathbf{C}_n)$. We also compute the filtering posterior $p(\mathbf{u}_n|\mathbf{y}_{1:n})$ by marginalising over the encoding, and unknown system parameters with Monte-Carlo; for full details, see Appendix C. We note also that this procedure can also be realised by other nonlinear filters, e.g., ensemble Kalman filters (Chen et al., 2022) or particle filters (Corenflos et al., 2021).

Choice of Variational Family. As with the decoder, encoder parameters ϕ are shared between variational distributions $\{q_\phi(\mathbf{x}_n|\mathbf{y}_n)\}_{n=1}^N$ to give an amortized approach (Kingma et al., 2019). Unless otherwise specified, for each n the encoding has the form $q_\phi(\mathbf{x}_n|\mathbf{y}_n) = \mathcal{N}(\mu_\phi(\mathbf{y}_n), \sigma_\phi(\mathbf{y}_n))$. Functions $\mu_\phi(\mathbf{y}_n) : \mathbb{R}^{n_y} \to \mathbb{R}^{n_x}$ and $\sigma_\phi(\mathbf{y}_n) : \mathbb{R}^{n_y} \to \mathbb{R}^{n_x}$ are neural networks, with parameters ϕ to be learnt. Specific encoding architectures are given in Appendix A. As for q_λ, we set it to a Gaussian with mean μ_λ and variance $\mathrm{diag}(\sigma_\lambda)$.

5 Experiments

We present three examples with different dynamical systems and data-generating processes. To demonstrate the generality of the method, the first uses the stochastic Lorenz-63 system (Lorenz, 1963), a highly nonlinear stochastic ODE. In this case, high-dimensional observations are of a velocity field being modulated by the chaotic system. For the final two examples, we use video data. We consider the advection and Korteweg-de Vries (KdV) PDEs, and, in these examples, we indirectly observe high-dimensional representations in the form of video data. This mimics the experimental setup of the various DVAE papers (e.g., Fraccaro et al., 2017; Pearce et al., 2018; Jazbec et al., 2021; Fortuin et al., 2020; Zhu et al., 2022; Girin et al., 2021). For these two examples, video datasets are generated in a similar fashion. In both cases, simulated data emulate the scenario where a noisy video of an internal wave profile is captured. Internal waves arise as waves of depression or elevation flowing within a density-stratified fluid at regions of maximum density gradient (Gerkema & Zimmerman, 2008). Our experiment setup is thus mimics an idealised setup where a black-and-white, side-on, video of a laboratory experiment has been obtained, and is inspired by scenarios where the high-resolution use of classical measurement devices is not feasible, yet the use of commonplace video-capturing devices is (see, e.g., Horn et al., 2001; 2002).

The advection equation example is motivated by an internal wave propagating undisturbed through some medium. For this linear case, comparisons with the KVAE reveal that after training, the Φ-DVAE outperforms both in terms of the estimated ELBO and in terms of the mean-squared-error (MSE). The KdV example is a more complex case, and extends into the nonlinear PDE setting, while also being a classical model for internal waves (see, e.g., Drazin & Johnson, 1989). For this example, comparisons are made with VRNNs, the GPVAE, and the standard VAE. We demonstrate that the MSE of the Φ-DVAE is comparable or better than these approaches. Furthermore, for joint state and parameter inference, we verify the methodology and demonstrate contraction of the posterior about the truth.

5.1 Lorenz-63 Example

In our first example, the latent dynamical model $p(\mathbf{u}_n|\mathbf{u}_{n-1}, \mathbf{\Lambda})$ is given by an Euler-Maruyama discretisation (Kloeden & Platen, 1992) of the stochastic Lorenz-63 system,

$$\begin{aligned} \mathrm{d}u_1 &= (-\sigma u_1 + \sigma u_2)\mathrm{d}t + \mathrm{d}w_1, \\ \mathrm{d}u_2 &= (-u_1 u_3 + r u_1 - u_2)\mathrm{d}t + \mathrm{d}w_2, \\ \mathrm{d}u_3 &= (u_1 u_2 - b u_3)\mathrm{d}t + \mathrm{d}w_3, \end{aligned} \tag{11}$$

where $\mathbf{u}(t) := [u_1(t), u_2(t), u_3(t)]^\top, t \in [0, 6]$, $\mathbf{u}_n = [u_1(n\Delta_t), u_2(n\Delta_t), u_3(n\Delta_t)]$, $\mathbf{\Lambda} = \{\sigma, r, b\}$, and w_1, w_2, and w_3 are independent Brownian motion processes (Øksendal, 2003; Särkkä & Solin, 2019). For full details we refer to Appendices A and C. The Lorenz-63 system is classical system widely used to benchmark filtering and data assimilation methods (Akyildiz & Míguez, 2020). It was popularised in Lorenz (1963) through its characterisation of "deterministic nonperiodic flow", and is a common example of chaotic dynamics.

We observe synthetic 2D velocity fields, $\mathbf{y}_{1:N}$, of convective fluid flow, and we use our method to embed these synthetic data into the stochastic Lorenz-63 system. The Lorenz-63 system is related to the velocity fields through a truncated spectral expansion (see, e.g., Wouters, 2013). In brief, it is assumed that the velocity fields have no vertical velocity, so the 3D velocity field is realised in 2D. The velocity field can be described by the stream function $\psi := \psi(s_1, s_2, t)$, where s_1 and s_2 are the spatial coordinates of variation, respectively, and thus:

$$\mathbf{y}(t) = (-\partial_{s_2}\psi, 0, \partial_{s_1}\psi)\,. \tag{12}$$

A truncated spectral approximation and a transform to non-dimensional equations yields $\psi(s_1, s_2, t) \propto u_1(t)\sin(\pi s_1/l)\sin(\pi s_2/d)$, where $u_1(t)$ is governed by the Lorenz-63 ODE (i.e., equation 11 with $w_i \equiv 0$).

To generate the synthetic data $\mathbf{y}_{1:N}$, we generate a trajectory $\mathbf{u}^{\text{true}}_{1:N}$ from the deterministic version of 11, at discrete timepoints $n\Delta_t$, and use the generated $u^{\text{true}}_{1,n}$ to compute the two-dimensional velocity field, $\mathbf{y}_n$, via $\psi(s_1, s_2, t)$. This corresponds to having a middle layer $x_n = u_{1,n} + w_n$ where $w_n \sim \mathcal{N}(0, R^2)$, with likelihood $p(x_n|\mathbf{u}_n) = \mathcal{N}(\mathbf{h}^\top \mathbf{u}_n, R^2)$ where $\mathbf{h} = [1, 0, 0]^\top$. The synthetic data of $\mathbf{u}^{\text{true}}(t)$, $\mathbf{x}_{1:N}$, and $\mathbf{y}_{1:N}$ are visualised in Figure 2; the full trajectory $\mathbf{u}^{\text{true}}(t)$ is shown in 3D, and the velocity field $\mathbf{y}_n$ is shown in 2D, for a single n. The decoding is assumed to be of the form $p_{\mathbf{w}}(\mathbf{y}_n|x_n) = \mathcal{N}(\mathbf{w}x_n, \eta^2\mathbf{I})$, with unknown coefficients $\mathbf{w} \in \mathbb{R}^{n_y}$. The variational encoding $q_{\mathbf{w}}(x_n|\mathbf{y}_n)$ is determined via a pseudo-inverse, as detailed in Appendix A, along with the relevant hyperparameters and numerical details.

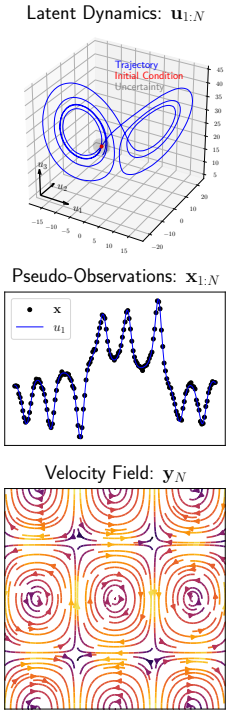
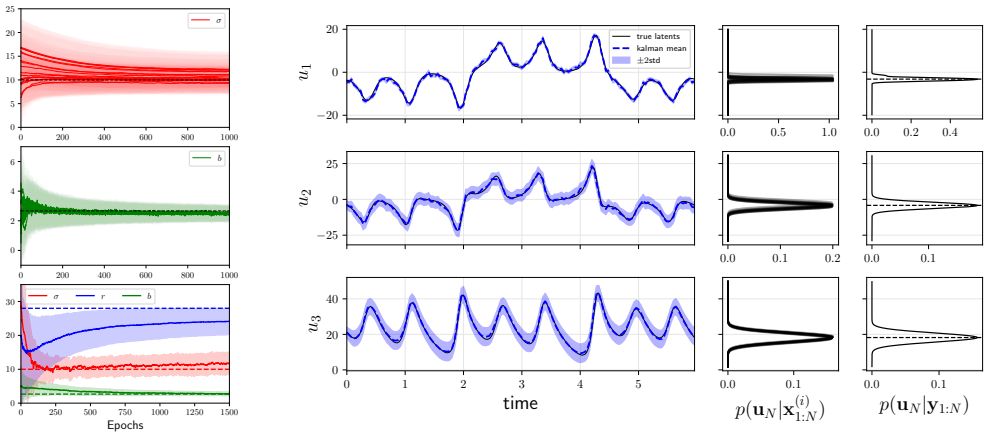


Figure 2: Lorenz-63: latent states $\mathbf{u}_{1:N}$, pseudo-observations $\mathbf{x}_{1:N}$, and velocity field $\mathbf{y}_N$.

Figure 3a displays parameter estimation results. For individual σ and b estimation (Figure 3a, top and center), we initialise the variational posterior randomly, and visualise each across training epochs. Both results show the posteriors contracting about the true value, with b demonstrating more rapid convergence, visually. For joint estimation (Figure 3a, bottom), similar behaviour is observed, with the parameter r not identified by the final epoch. We conjecture this is due to identifiability with other parameters when estimating jointly. Note, however, that the true values are all contained within the confidence bands of the final variational posteriors.



(a) Parameter estimation

(b) Inference

Figure 3: Lorenz-63 system: (a) the results of parameter estimation for (top) σ estimation, (center) b estimation, and (bottom) joint estimation of $\mathbf{\Lambda} = \{\sigma, r, b\}$. In (b) we show an example result of state estimation. Left column is the true Lorenz states vs. ExKF means, center and right columns show the distribution of the estimate of the final states.

We also investigate the posterior inference achieved by Φ-DVAE. We visualise the filtering posterior through time, conditioned on a sample from the trained encoder $\mathbf{x}_{1:N}^{(i)} \sim q_\phi(\cdot|\mathbf{y}_{1:N})$, with fixed, known $\mathbf{\Lambda}$. Figure 3b (left) shows clear agreement between the filter mean and the latent states $\mathbf{u}_{1:N}^{\text{true}}$. Marginalising over the encoding (Figure 3b (centre)) targets the filtering posterior directly conditioned on observed data $\mathbf{y}_{1:N}$, and demonstrates unbiased mean estimates of the true state. This is particularly clear for the first latent dimension, where the posterior conditioned on an individual sample $\mathbf{x}_{1:N}^{(i)}$ often has poor posterior coverage of the true value (cf. Figure 3b (right)).

5.2 Advection PDE

As a second example, we consider the advection equation with periodic boundary conditions. In this example, we derive the transition density $p(\mathbf{u}_n|\mathbf{u}_{n-1}, \mathbf{\Lambda})$ from a statFEM discretisation of a stochastic advection equation:

$$\partial_t u + c\,\partial_s u = \dot{\xi}, \quad \dot{\xi} \sim \mathcal{GP}(0, \delta(t-t') \cdot k(s,s')), \tag{13}$$

where $u := u(s,t)$, $\xi := \xi(s,t)$, $s \in [0,1]$, $t \in [0,40]$, and $u(s,t) = u(s+1,t)$. Recall that, as in Section 5.2, the FEM coefficients $\mathbf{u}_n = (u_1(n\Delta_t), \ldots, u_{n_u}(n\Delta_t))$ are the latent variables. These are related to the discretised solution via $u_h(s, n\Delta_t) = \sum_{i=1}^{n_u} u_i(n\Delta_t)\phi_i(s)$ (see Appendix A and C).

Video data $\mathbf{y}_{1:N}$ is generated from the deterministic version of equation 13 (i.e. equation 13 with $\xi \equiv 0$). A trajectory $\mathbf{u}_{1:N}^{\text{true}}$ is simulated and the corresponding FEM solutions $u_h^{\text{true}}(s, n\Delta_t)$ are imposed onto a 2D grid. On the grid, pixels below $u_h^{\text{true}}(s, n\Delta_t)$ are lit-up in a binary fashion, with salt-and-peper noise (Gonzalez & Woods, 2007). In this experiment we use fixed parameters, setting $\mathbf{\Lambda} \equiv c = 0.5$. We set the decoder as $p_\theta(\mathbf{y}_n|\mathbf{x}_n) = \text{Bern}(\mu_\theta(\mathbf{x}_n))$ and the encoder as $q_\phi(\mathbf{x}_n|\mathbf{y}_n) = \mathcal{N}(\mu_\phi(\mathbf{y}_n), \sigma_\phi(\mathbf{y}_n))$. As previous, see Appendix A for full details.

Due to linearity of the underlying dynamical system, we compare the Φ-DVAE to the KVAE for a set of video data generated from the advection equation, for various dimensions of the KVAE latent space. Specifying a particular form of latent dynamics on the latent states increases the inductive bias imposed on the latent space, and should provide faster learning — and more likely representations — than with learnt dynamics.

To explore this, we compare our method to KVAE for the linear advection example in terms of the ELBO and normalised MSE, over training epochs (all methods use Adam (Kingma & Ba, 2017)). These are plotted in Figure 4. For the MSE, KVAE quickly learns to reconstruct the images, with Φ-DVAE taking longer to reconstruct with similar accuracy but eventually producing better reconstructions (final MSEs 0.0221 vs. 0.0533 for the Φ-DVAE, KVAE-64, respectively). The ELBO for Φ-DVAE is rapidly optimised — the inductive bias forces learned representations of the data $\mathbf{y}_{1:N}$ to accord with the well-specified latent dynamic model. The final trained Φ-DVAE has learnt highly probable latent encodings $p(\mathbf{x}_{1:N}|\mathbf{\Lambda})$, whilst still providing accurate reconstruction $p_\theta(\mathbf{y}_{1:N}|\mathbf{x}_{1:N})$.

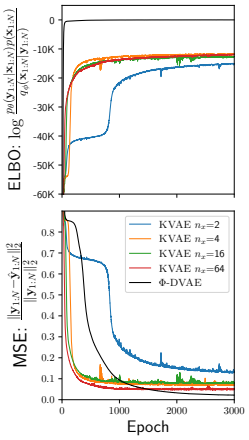


Figure 4: Advection PDE: Comparison of the ELBO (*top*) and MSE (*bottom*) of reconstruction compared with ground truth images, for (Φ-DVAE) against KVAE.

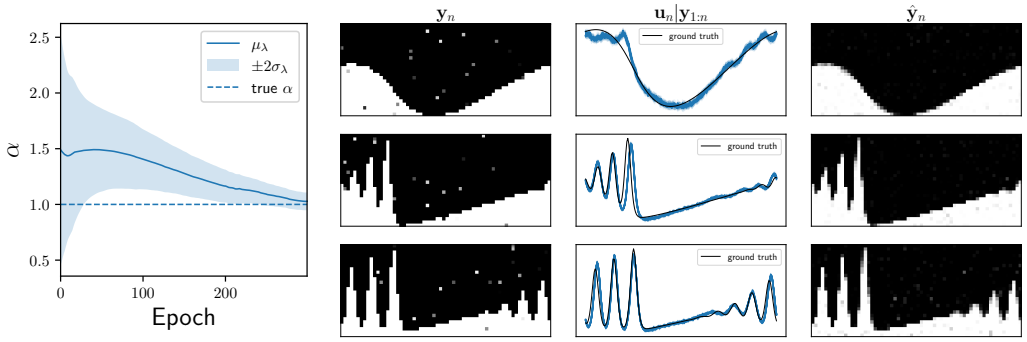
5.3 Korteweg–de Vries PDE

Our final example uses the KdV equation as the underlying dynamical system. As previously, the latent transition density $p(\mathbf{u}_n|\mathbf{u}_{n-1}, \mathbf{\Lambda})$ defines the evolution of the FEM coefficients, as given by a statFEM discretisation of a stochastic KdV equation:

$$\partial_t u + \alpha u \partial_s u + \beta \partial_s^3 u = \dot{\xi}, \quad \dot{\xi} \sim \mathcal{GP}(0, \delta(t-t') \cdot k(s,s')),$$

where $u := u(s,t)$, $\xi := \xi(s,t)$, $s \in [0,2]$, $t \in [0,1]$, and $u(s,t) = u(s+2,t)$. Parameters are $\mathbf{\Lambda} = \{\alpha, \beta\}$.

Data is simulated in the same fashion as in the advection equation: we simulate a trajectory $\mathbf{u}_{1:N}^{\text{true}}$ using a finite element method (FEM) discretisation of the deterministic KdV equation and we impose



(a) Parameter estimation. (b) Image data. (c) Latent inference. (d) Reconstructions.

Figure 6: KDV joint inference results; frames shown for $n = \{10, 50, 75\}$.

FEM solutions $u_h^{\text{true}}(s, n\Delta_t)$ on a 2D grid. We light up pixels below the solution, and add salt-and-pepper noise. For the data-generating process, we use the parameters $\mathbf{\Lambda} = \{\alpha = 1, \beta = 0.022^2\}$, and the initial condition of $u_h^{\text{true}}(s, 0) = \cos(\pi s)$ as in the classical work of Zabusky & Kruskal (1965). This regime is characterised by the steepening of the initial condition and the generation of solitons; nonlinear waves which have particle-like interactions (Drazin & Johnson, 1989). As for the advection example, we set the decoder as $p_\theta(\mathbf{y}_n|\mathbf{x}_n) = \text{Bern}(\mu_\theta(\mathbf{x}_n))$ and the encoder as $q_\phi(\mathbf{x}_n|\mathbf{y}_n) = \mathcal{N}(\mu_\phi(\mathbf{y}_n), \sigma_\phi(\mathbf{y}_n))$; again, see Appendix A for details.

We test the VRNN, GPVAE and the standard VAE approaches alongside Φ-DVAE with equal encoding dimension $n_x = 40$. We report the normalised MSE after a fixed number of epochs, using Adam (Kingma & Ba, 2017) with the same learning rate for each method. Φ-DVAE outperforms both GPVAE and the standard VAE in terms of median MSE. Note the variation, in MSE, of the standard VAE is also greater than the other models, suggesting that the dynamical structure provides more consistent learning. Both Φ-DVAE and VRNN perform well, producing visually similar reconstructions, with the VRNN having lower median MSE (0.0239 vs 0.0105, respectively).

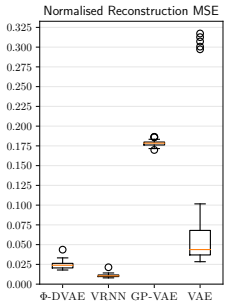


Figure 5: Normalised MSE of reconstruction after 200 epochs, for 30 independent simulations.

We report joint estimation results for the partially known KDV PDE, where we fix $\beta = 0.022^2$, and estimate α. The prior over α, $p(\alpha) = \mathcal{N}(1.5, 0.3^2)$, is semi-informative. Joint inference of α and latent states $\mathbf{u}_n$ is shown in Figure 6. The Gaussian variational posterior, $q_\lambda(\alpha) = \mathcal{N}(\mu_\lambda, \sigma_\lambda^2)$ (initialised at the prior), contracts about the true value $\alpha = 1.0$ (see Figure 6a). The filtering posterior, $p(\mathbf{u}_n|\mathbf{y}_{1:n})$, is shown in Figure 6c. This is obtained via Monte Carlo approximation, marginalising over the variational posteriors q_ϕ, q_λ, to account for uncertainty in the encoding and parameter estimates. Including a structured prior on the latent space has forced the encoding to be representative of observations taken from the KDV system, clearly capturing the latent dynamics causing the variation in the image data. Figures 6b and 6d display the image data and reconstructions respectively, showing the ability of Φ-DVAE to both accurately reconstruct and de-noise the data.

6 Conclusion

In this paper we developed Φ-DVAE, a methodology that allows for the incorporation of unstructured data into physical models, in settings where the model-data mapping may be unknown. The proposed approach utilizes variational autoencoders and nonlinear filtering algorithms for PDEs, to learn physically interpretable latent spaces where analysis and prediction can be performed straightforwardly. Our framework connects traditional nonlinear filtering techniques and VAEs, opening up the possibility of further combinations of these methods. Future work will focus on more challenging PDE systems, as well as more complex, and higher-dimensional, observational data.

REPRODUCIBILITY STATEMENT

We have included full numerical details to reproduce all results in our paper in Appendix A. The code we have developed for this paper will be made publicly available after the decision, enabling the generation of all datasets and figures.

REFERENCES

Ömer Deniz Akyildiz and Joaquín Míguez. Nudging the particle filter. *Statistics and Computing*, 30 (2):305–330, 2020.

Ömer Deniz Akyildiz, Connor Duffin, Sotirios Sabanis, and Mark Girolami. Statistical Finite Elements via Langevin Dynamics. *SIAM/ASA Journal of Uncertainty Quantification*, 2022.

Brian D.O. Anderson and John B. Moore. *Optimal filtering*. Englewood Cliffs, N.J. Prentice Hall, 1979.

Justin Bayer and Christian Osendorfer. Learning stochastic recurrent networks. *arXiv preprint arXiv:1411.7610*, 2014.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 112(518):859–877, April 2017. ISSN 0162-1459. doi: 10.1080/01621459.2017.1285773.

Marc Bocquet and Pavel Sakov. Joint state and parameter estimation with an iterative ensemble kalman smoother. *Nonlinear Processes in Geophysics*, 20(5):803–818, 2013.

G. E. P. Box. Robustness in the Strategy of Scientific Model Building. In ROBERT L. Launer and GRAHAM N. Wilkinson (eds.), *Robustness in Statistics*, pp. 201–236. Academic Press, January 1979. ISBN 978-0-12-438150-6. doi: 10.1016/B978-0-12-438150-6.50018-2.

Susanne C. Brenner and L. Ridgway Scott. *The Mathematical Theory of Finite Element Methods*, volume 15 of *Texts in Applied Mathematics*. Springer New York, New York, NY, 2008. ISBN 978-0-387-75933-3 978-0-387-75934-0. doi: 10.1007/978-0-387-75934-0.

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

Kathleen Champion, Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Data-driven discovery of coordinates and governing equations. *Proceedings of the National Academy of Sciences*, 116(45): 22445–22451, 2019.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural Ordinary Differential Equations. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Yuming Chen, Daniel Sanz-Alonso, and Rebecca Willett. Autodifferentiable ensemble kalman filters. *SIAM Journal on Mathematics of Data Science*, 4(2):801–833, 2022.

Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. *Advances in neural information processing systems*, 28, 2015.

Adrien Corenflos, James Thornton, George Deligiannidis, and Arnaud Doucet. Differentiable particle filtering via entropy-regularized optimal transport. In *International Conference on Machine Learning*, pp. 2100–2111. PMLR, 2021.

Giuseppe Da Prato and Jerzy Zabczyk. *Stochastic Equations in Infinite Dimensions*. Encyclopedia of Mathematics and Its Applications. Cambridge University Press, Cambridge, second edition, 2014. ISBN 978-1-107-05584-1. doi: 10.1017/CBO9781107295513.

Arnaud Debussche and Jacques Printems. Numerical simulation of the stochastic Korteweg–de Vries equation. *Physica D: Nonlinear Phenomena*, 134(2):200–226, October 1999. ISSN 0167-2789. doi: 10.1016/S0167-2789(99)00072-X.

Susanne Ditlevsen and Adeline Samson. Estimation in the partially observed stochastic morris–lecar neuronal model with particle filter and stochastic approximation methods. *The annals of applied statistics*, 8(2):674–702, 2014.

Denis Dochain. State and parameter estimation in chemical and biochemical processes: a tutorial. *Journal of process control*, 13(8):801–818, 2003.

Arnaud Doucet, Simon Godsill, and Christophe Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10(3):197–208, July 2000. ISSN 1573-1375. doi: 10.1023/A:1008935410038.

P. G. Drazin and R. S. Johnson. *Solitons: An Introduction*. Cambridge Texts in Applied Mathematics. Cambridge University Press, Cambridge, second edition, 1989. ISBN 978-0-521-33655-0. doi: 10.1017/CBO9781139172059.

Connor Duffin, Edward Cripps, Thomas Stemler, and Mark Girolami. Statistical finite elements for misspecified models. *Proceedings of the National Academy of Sciences*, 118(2), 2021.

Connor Duffin, Edward Cripps, Thomas Stemler, and Mark Girolami. Low-rank statistical finite elements for scalable model-data synthesis. *Journal of Computational Physics*, 463, August 2022. ISSN 0021-9991. doi: 10.1016/j.jcp.2022.111261.

N. Benjamin Erichson, Michael Muehlebach, and Michael W. Mahoney. Physics-informed Autoencoders for Lyapunov-stable Fluid Flow Prediction, May 2019.

Lawrence Evans. *Partial Differential Equations*, volume 19 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, Rhode Island, second edition, March 2010. ISBN 978-0-8218-4974-3 978-1-4704-1144-2. doi: 10.1090/gsm/019.

Geir Evensen. The Ensemble Kalman Filter: Theoretical formulation and practical implementation. *Ocean Dynamics*, 53(4):343–367, November 2003. ISSN 1616-7228. doi: 10.1007/s10236-003-0036-9.

Vincent Fortuin, Dmitry Baranchuk, Gunnar Rätsch, and Stephan Mandt. Gp-vae: Deep probabilistic time series imputation. In *International conference on artificial intelligence and statistics*, pp. 1651–1661. PMLR, 2020.

Marco Fraccaro, Simon Kamronn, Ulrich Paquet, and Ole Winther. A Disentangled Recognition and Nonlinear Dynamics Model for Unsupervised Learning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

T Gerkema and J T F Zimmerman. An introduction to internal waves. *Lecture Notes, Royal NIOZ, Texel*, 207:207, 2008.

Craig Gin, Bethany Lusch, Steven L. Brunton, and J. Nathan Kutz. Deep learning models for global coordinate transformations that linearise PDEs. *European Journal of Applied Mathematics*, 32(3): 515–539, June 2021. ISSN 0956-7925, 1469-4425. doi: 10.1017/S0956792520000327.

Laurent Girin, Simon Leglaive, Xiaoyu Bie, Julien Diard, Thomas Hueber, and Xavier Alameda-Pineda. Dynamical variational autoencoders: A comprehensive review. *Foundations and Trends in Machine Learning*, 15(1-2):1–175, 2021.

Mark Girolami, Eky Febrianto, Ge Yin, and Fehmi Cirak. The statistical finite element method (statFEM) for coherent synthesis of observation data and model predictions. *Computer Methods in Applied Mechanics and Engineering*, 375:113533, March 2021. ISSN 0045-7825. doi: 10.1016/j.cma.2020.113533.

Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Pearson, Upper Saddle River, N.J, 3rd edition edition, August 2007. ISBN 978-0-13-168728-8.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning Latent Dynamics for Planning from Pixels. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 2555–2565. PMLR, May 2019.

Jouni Hartikainen and Simo Sarkka. Kalman filtering and smoothing solutions to temporal Gaussian process regression models. In *2010 IEEE International Workshop on Machine Learning for Signal Processing*, pp. 379–384, Kittila, Finland, August 2010. IEEE. ISBN 978-1-4244-7875-0. doi: 10.1109/MLSP.2010.5589113.

Carlos X. Hernández, Hannah K. Wayment-Steele, Mohammad M. Sultan, Brooke E. Husic, and Vijay S. Pande. Variational encoding of complex dynamics. *Physical Review E*, 97(6):062412, June 2018. ISSN 2470-0045, 2470-0053. doi: 10.1103/PhysRevE.97.062412.

D. A. Horn, J. Imberger, and G. N. Ivey. The degeneration of large-scale interfacial gravity waves in lakes. *Journal of Fluid Mechanics*, 434:181–207, May 2001. ISSN 0022-1120, 1469-7645. doi: 10.1017/S0022112001003536.

D. A. Horn, J. Imberger, G. N. Ivey, and L. G. Redekopp. A weakly nonlinear model of long internal waves in closed basins. *Journal of Fluid Mechanics*, 467:269–287, September 2002. ISSN 1469-7645, 0022-1120. doi: 10.1017/S0022112002001362.

Metod Jazbec, Matt Ashman, Vincent Fortuin, Michael Pearce, Stephan Mandt, and Gunnar Rätsch. Scalable Gaussian Process Variational Autoencoders. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, pp. 3511–3519. PMLR, March 2021.

Andrew H Jazwinski. *Stochastic Processes and Filtering Theory*. Academic Press, 1970.

K Judd and L Smith. Indistinguishable states II. The imperfect model scenario. *Physica D: Nonlinear Phenomena*, 196(3-4):224–242, September 2004. ISSN 01672789. doi: 10.1016/S0167-2789(04)00182-4.

R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, March 1960. ISSN 0021-9223. doi: 10.1115/1.3662552.

Nikolas Kantas, Arnaud Doucet, Sumeetpal S Singh, Jan Maciejowski, and Nicolas Chopin. On particle methods for parameter estimation in state-space models. *Statistical science*, 30(3):328–351, 2015.

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

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, January 2017.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes, May 2014.

Diederik P Kingma, Max Welling, et al. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019.

Peter Kloeden and Eckhard Platen. *Numerical Solution of Stochastic Differential Equations*. Applications of Mathematics. Springer Berlin Heidelberg, 1992.

Costas Kravaris, Juergen Hahn, and Yunfei Chu. Advances and selected recent developments in state and parameter estimation. *Computers & chemical engineering*, 51:111–123, 2013.

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

Kody Law, Andrew Stuart, and Konstantinos Zygalakis. *Data Assimilation: A Mathematical Introduction*, volume 62. Springer, Cham, Switerland, 2015. ISBN 978-3-319-20325-6.

Ryan Lopez and Paul J. Atzberger. Variational Autoencoders for Learning Nonlinear Dynamics of Physical Systems, March 2021.

Edward N Lorenz. Deterministic nonperiodic flow. *Journal of atmospheric sciences*, 20(2):130–141, 1963.

Peter Y. Lu, Samuel Kim, and Marin Soljačić. Extracting Interpretable Physical Parameters from Spatiotemporal Systems Using Unsupervised Learning. *Physical Review X*, 10(3):031056, September 2020. doi: 10.1103/PhysRevX.10.031056.

Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature communications*, 9(1):1–10, 2018.

Hamid Moradkhani, Soroosh Sorooshian, Hoshin V Gupta, and Paul R Houser. Dual state–parameter estimation of hydrological models using ensemble kalman filter. *Advances in water resources*, 28 (2):135–147, 2005.

Jeremy Morton, Antony Jameson, Mykel J Kochenderfer, and Freddie Witherden. Deep Dynamical Modeling and Control of Unsteady Fluid Flows. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Bernt Øksendal. Stochastic differential equations. In *Stochastic Differential Equations*, pp. 65–84. Springer, 2003.

Samuel E. Otto and Clarence W. Rowley. Linearly Recurrent Autoencoder Networks for Learning Dynamics. *SIAM Journal on Applied Dynamical Systems*, 18(1):558–593, January 2019. doi: 10.1137/18M1177846.

Michael Pearce. The gaussian process prior vae for interpretable latent dynamics from pixels. In *Symposium on advances in approximate bayesian inference*, pp. 1–12, 2020. tex.organization: PMLR.

Michael Pearce, Silvia Chiappa, and Ulrich Paquet. Comparing interpretable inference models for videos of physical motion. In *1st symposium on advances in approximate bayesian inference*, 2018.

Sebastian Reich and Colin Cotter. *Probabilistic forecasting and Bayesian data assimilation*. Cambridge University Press, 2015.

Simo Särkkä and Arno Solin. *Applied Stochastic Differential Equations*, volume 10. Cambridge University Press, 2019.

W. E. Schiesser. *The Numerical Method of Lines: Integration of Partial Differential Equations*. Academic Press, 1991. ISBN 978-0-12-624130-3.

Geir Storvik. Particle filters for state-space models with the presence of unknown static parameters. *IEEE Transactions on signal Processing*, 50(2):281–289, 2002.

A. M. Stuart. Inverse problems: A Bayesian perspective. *Acta Numerica*, 19:451–559, May 2010. ISSN 0962-4929, 1474-0508. doi: 10.1017/S0962492910000061.

Naoya Takeishi, Yoshinobu Kawahara, and Takehisa Yairi. Learning Koopman Invariant Subspaces for Dynamic Mode Decomposition. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Albert Tarantola. *Inverse Problem Theory and Methods for Model Parameter Estimation*. Society for Industrial and Applied Mathematics, January 2005. ISBN 978-0-89871-572-9 978-0-89871-792-1. doi: 10.1137/1.9780898717921.

Vidar Thomée. *Galerkin Finite Element Methods for Parabolic Problems*. Number v. 25 in Springer Series in Computational Mathematics. Springer, Berlin ; New York, 2nd ed edition, 2006. ISBN 978-3-540-33121-6.

Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to Control: A Locally Linear Latent Dynamics Model for Control from Raw Images. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

Christopher KI Williams and Carl Edward Rasmussen. *Gaussian Processes for Machine Learning*, volume 2. MIT press Cambridge, MA, 2006.

Jeroen Wouters. A brief introduction to the Lorenz'63 system. 2013.

Cagatay Yildiz, Markus Heinonen, and Harri Lahdesmaki. ODE2VAE: Deep generative second order ODEs with Bayesian neural networks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

N. J. Zabusky and M. D. Kruskal. Interaction of "Solitons" in a Collisionless Plasma and the Recurrence of Initial States. *Physical Review Letters*, 15(6):240–243, August 1965. doi: 10.1103/PhysRevLett.15.240.

Harrison Zhu, Carles Balsells Rodas, and Yingzhen Li. Markovian Gaussian Process Variational Autoencoders, July 2022.

A NUMERICAL DETAILS AND NETWORK ARCHITECTURES

Lorenz-63 experiment.

- Time-series length: $N = 150$.
- Input: $n_y = 200$.
- Pseudo-observations: $n_x = 1$.
- Latent $n_u = 3$.
- Decoder: $p_{\mathbf{w}}(\mathbf{y}_n|x_n) = \mathcal{N}(\mathbf{w}x_n, \eta^2\mathbf{I})$, $\eta = 0.005$.
- Encoder: $q_{\mathbf{w}}(x_n|\mathbf{y}_n) = \mathcal{N}((\mathbf{w}^\top\mathbf{w})^{-1}\mathbf{w}^\top\mathbf{y}_n, \eta^2(\mathbf{w}^\top\mathbf{w})^{-1})$
- Latent initial condition: $\mathbf{u}_0 = [-3.7277, -3.8239, 21.1507]^\top$
- Latent noise processes: $\mathbf{L} = \text{diag}(0.2^2)$, $\mathbf{R} = \text{diag}(0.4^2)$.
- Latent discretisation: Euler-Maruyama, $dt = 0.001$.
- Joint parameter prior: $p([\sigma, r, b]) = \mathcal{N}([30, 20, 5]^\top, \text{diag}([12^2, 10^2, 3^2]^\top))$.
- Optimiser: Adam, learning rate = 10^{-4}.

To generate the data, we simulate the Lorenz SDE with $dt = 0.001$ and take pseudo-observations x_n every 40 time-steps of the latent system, for a total of $N = 150$ with $\Delta_t = 0.04$. Velocity measurements are taken in the s_1 and s_2 directions over a regular 10×10 grid on the domain $s_1, s_2 \in [-4, 4]$, via the streamfunction $\psi(s_1, s_2, t)$. These measurements are flattened to the data vector $\mathbf{y}_n \in \mathbb{R}^{n_y}$, $n_y = 200$. Parameters for data generation are $\mathbf{\Lambda} = \{\sigma = 10, r = 28, b = 8/3\}$.

Advection equation experiment.

- Time-series length: $N = 200$.
- Input: $n_y = 784$.
- Pseudo-observations: $n_x = 64$.
- Latent $n_u = 64$.
- Decoder: $p_\theta(\mathbf{y}_n|\mathbf{x}_n) = \text{Bern}(\mu_\theta(\mathbf{x}_n))$
- $\mu_\theta(\cdot)$: MLP, two fully connected hidden layers with dimension 128
- Encoder: $q_\phi(\mathbf{x}_n|\mathbf{y}_n) = \mathcal{N}(\mu_\phi(\mathbf{y}_n), \sigma_\phi(\mathbf{y}_n))$
- $\mu_\phi(\cdot), \sigma_\phi(\cdot)$: MLP, two fully connected hidden layers with dimension 128
- Neural Network activations: LeakyReLU, negative slope = 0.01
- Latent initial condition: $u(s, 0) = \exp(-(x - 2.5)^2/0.1)$.
- Latent noise processes: $\rho = 0.02$, $\ell = 0.1$, $\mathbf{R} = \text{diag}(0.1^2)$.
- Latent discretisation: FEM, $C^0([0, 1])$ polynomial trial/test functions, Crank-Nicolson time discretisation, $dt = 0.02$.
- Optimiser: Adam, learning rate = 0.001.

To generate the data, we simulate the advection equation with $dt = 0.02$, observing every 10 time-steps for $\Delta_t = 0.2$ and $N = 200$. Latent dimensions are $n_u = 64$ and $n_x = 64$, with each image 28×28 pixels. These images are then flattened to vectors $\mathbf{y}_n \in [0, 1]^{n_y}$, with $n_y = 784$.

KDV equation experiment.

- Time-series length: $N = 100$.
- Input: $n_y = 1792$.
- Pseudo-observations: $n_x = 40$.
- Latent $n_u = 600$.
- Decoder: $p_\theta(\mathbf{y}_n|\mathbf{x}_n) = \text{Bern}(\mu_\theta(\mathbf{x}_n))$

- $\mu_\theta(\cdot)$: MLP, two fully connected hidden layers with dimension 128
- Encoder: $q_\phi(\mathbf{x}_n|\mathbf{y}_n) = \mathcal{N}(\mu_\phi(\mathbf{y}_n), \sigma_\phi(\mathbf{y}_n))$
- $\mu_\phi(\cdot), \sigma_\phi(\cdot)$: MLP, two fully connected hidden layers with dimension 128
- Neural Network activations: LeakyReLU, negative slope $= 0.01$
- Latent initial condition: $u(s, 0) = \cos(\pi s)$.
- Latent noise processes: $\rho = 0.01$, $\ell = 0.2$, $\mathbf{R} = \mathrm{diag}(0.05^2)$.
- Latent discretisation: Petrov-Galerkin approach of Debussche & Printems (1999): $C^0([0, 2])$ polynomial trial functions, Crank-Nicolson time discretisation, $dt = 0.01$.
- Parameter prior: $p(\alpha) = \mathcal{N}(1.5, 0.3^2)$.
- Optimiser: Adam, learning rate = 0.005.

To generate the data we simulate the KDV equation with $dt = 0.01$, observing every timestep for $\Delta_t = 0.01$, and we take $N = 100$ observations $\mathbf{y}_n$, with $\mathbf{y}_n \in [0, 1]^{n_y}$. Each $\mathbf{y}_n$ is a flattened image of dimension $n_y = 64 \times 28 = 1792$, and we encode to pseudo-observations $\mathbf{x}_n$ of dimension $n_x = 40$. The latent state dimension is $n_u = 600$.

Linear decoding/encoding. If a linear data generation is assumed from $\mathbf{x}_{1:N}$ to $\mathbf{y}_{1:N}$, then this structure can inform decoding. With a linear decoder of the form:

$$p_{\mathbf{A}}(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{A}\mathbf{x}, \eta^2 I).$$

In this case, we use the "inverted" linear decoder given by:

$$q_{\mathbf{A}}(\mathbf{x}|\mathbf{y}) = \mathcal{N}((\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\mathbf{y}, \eta^2(\mathbf{A}^\top\mathbf{A})^{-1}).$$

By selecting the encoder appropriately, the space of parameterised variational distributions can be restricted to align with our beliefs about the data generation process.

B Full Variational Framework

We derive the approximate ELBO for joint estimation of dynamic parameters $\mathbf{\Lambda}$, and autoencoder parameters ϕ, θ. We start by writing the evidence

$$p(\mathbf{y}_{1:N}) = \int p(\mathbf{u}_{1:N}, \mathbf{x}_{1:N}, \mathbf{\Lambda}, \mathbf{y}_{1:N}) \mathrm{d}\mathbf{x}_{1:N} \mathrm{d}\mathbf{u}_{1:N} \mathrm{d}\mathbf{\Lambda}.$$

We maximize $\log p(\mathbf{y}_{1:N})$ as

$$\begin{aligned}
\log p(\mathbf{y}_{1:N}) &= \log \int p(\mathbf{u}_{1:N}, \mathbf{x}_{1:N}, \mathbf{\Lambda}, \mathbf{y}_{1:N}) \mathrm{d}\mathbf{x}_{1:N} d\mathbf{u}_{1:N} \mathrm{d}\mathbf{\Lambda} \\
&= \log \int \frac{p(\mathbf{u}_{1:N}, \mathbf{x}_{1:N}, \mathbf{\Lambda}, \mathbf{y}_{1:N})}{q(\mathbf{u}_{1:N}, \mathbf{x}_{1:N}, \mathbf{\Lambda}|\mathbf{y}_{1:N})} q(\mathbf{u}_{1:N}, \mathbf{x}_{1:N}, \mathbf{\Lambda}|\mathbf{y}_{1:N}) \mathrm{d}\mathbf{x}_{1:N} \mathrm{d}\mathbf{u}_{1:N} \mathrm{d}\mathbf{\Lambda} \\
&\geq \int \log \left[\frac{p(\mathbf{u}_{1:N}, \mathbf{x}_{1:N}, \mathbf{\Lambda}, \mathbf{y}_{1:N})}{q(\mathbf{u}_{1:N}, \mathbf{x}_{1:N}, \mathbf{\Lambda}|\mathbf{y}_{1:N})} \right] q(\mathbf{u}_{1:N}, \mathbf{x}_{1:N}, \mathbf{\Lambda}|\mathbf{y}_{1:N}) \mathrm{d}\mathbf{x}_{1:N} \mathrm{d}\mathbf{u}_{1:N} \mathrm{d}\mathbf{\Lambda} \\
&= \text{ELBO},
\end{aligned}$$

where the third line follows from the application of Jensen's inequality. Our generative model determines the factorisation of the joint distribution, which is based on equation 1, equation 2, equation 3, and equation 4:

$$\begin{aligned}
p(\mathbf{u}_{1:N}, \mathbf{x}_{1:N}, \mathbf{\Lambda}, \mathbf{y}_{1:N}) &= p(\mathbf{y}_{1:N}|\mathbf{x}_{1:N}, \mathbf{u}_{1:N}, \mathbf{\Lambda}) p(\mathbf{x}_{1:N}|\mathbf{u}_{1:N}, \mathbf{\Lambda}) p(\mathbf{u}_{1:N}|\mathbf{\Lambda}) p(\mathbf{\Lambda}) \\
&= p_\theta(\mathbf{y}_{1:N}|\mathbf{x}_{1:N}) p(\mathbf{x}_{1:N}|\mathbf{u}_{1:N}, \mathbf{\Lambda}) p(\mathbf{u}_{1:N}|\mathbf{\Lambda}) p(\mathbf{\Lambda}).
\end{aligned}$$

Next, we plug this factorised distribution into the ELBO and obtain

$$\begin{aligned}
\text{ELBO} = \int \log \left[\frac{p_\theta(\mathbf{y}_{1:N}|\mathbf{x}_{1:N}) p(\mathbf{x}_{1:N}|\mathbf{u}_{1:N}, \mathbf{\Lambda}) p(\mathbf{u}_{1:N}|\mathbf{\Lambda}) p(\mathbf{\Lambda})}{q(\mathbf{u}_{1:N}, \mathbf{x}_{1:N}, \mathbf{\Lambda}|\mathbf{y}_{1:N})} \right] \\
\times q(\mathbf{u}_{1:N}, \mathbf{x}_{1:N}, \mathbf{\Lambda}|\mathbf{y}_{1:N}) \, \mathrm{d}\mathbf{x}_{1:N} \mathrm{d}\mathbf{u}_{1:N} \mathrm{d}\mathbf{\Lambda}.
\end{aligned}$$

The family of distributions which we use to approximate the posterior is described below. We assume a factorisation based on the model into variational encoding $q_\phi(\cdot)$, a full latent state posterior $q_\nu(\cdot)$, and the variational approximation to the parameter posterior $q_\lambda(\cdot)$:

$$q(\mathbf{u}_{1:N}, \mathbf{x}_{1:N}, \mathbf{\Lambda}|\mathbf{y}_{1:N}) = q_\nu(\mathbf{u}_{1:N}|\mathbf{x}_{1:N}, \mathbf{\Lambda})q_\lambda(\mathbf{\Lambda})q_\phi(\mathbf{x}_{1:N}|\mathbf{y}_{1:N}) \tag{14}$$

$$= q_\nu(\mathbf{u}_{1:N}|\mathbf{x}_{1:N}, \mathbf{\Lambda})q_\lambda(\mathbf{\Lambda}) \prod_{n=1}^{N} q_\phi(\mathbf{x}_n|\mathbf{y}_n). \tag{15}$$

The second line demonstrates the amortized structure of the autoencoder, where the same encoding parameters are shared across datapoints. We can then substitute this expression into our ELBO and obtain

$$\text{ELBO} = \int \log \left[\frac{p_\theta(\mathbf{y}_{1:N}|\mathbf{x}_{1:N})p(\mathbf{x}_{1:N}|\mathbf{u}_{1:N}, \mathbf{\Lambda})p(\mathbf{u}_{1:N}|\mathbf{\Lambda})p(\mathbf{\Lambda})}{q_\nu(\mathbf{u}_{1:N}|\mathbf{x}_{1:N}, \mathbf{\Lambda})q_\lambda(\mathbf{\Lambda})q_\phi(\mathbf{x}_{1:N}|\mathbf{y}_{1:N})} \right] \times q(\mathbf{u}_{1:N}, \mathbf{x}_{1:N}, \mathbf{\Lambda}|\mathbf{y}_{1:N}) \mathrm{d}\mathbf{x}_{1:N} \mathrm{d}\mathbf{u}_{1:N} \mathrm{d}\mathbf{\Lambda}.$$

Assuming the variational posterior is the exact filtering posterior, i.e., $q_\nu(\mathbf{u}_{1:N}|\mathbf{x}_{1:N}, \mathbf{\Lambda}) = p(\mathbf{u}_{1:N}|\mathbf{x}_{1:N}, \mathbf{\Lambda})$ then applying Bayes rule

$$\frac{p(\mathbf{x}_{1:N}|\mathbf{u}_{1:N}, \mathbf{\Lambda})p(\mathbf{u}_{1:N}|\mathbf{\Lambda})}{q_\nu(\mathbf{u}_{1:N}|\mathbf{x}_{1:N}, \mathbf{\Lambda})} = p(\mathbf{x}_{1:N}|\mathbf{\Lambda})$$

leads to a simplification of ELBO in terms of the marginal likelihood $p(\mathbf{x}_{1:N}|\mathbf{\Lambda})$ of the state-space model. Substituting this expression leads to

$$\begin{aligned}
\text{ELBO} &= \int \log \left[\frac{p_\theta(\mathbf{y}_{1:N}|\mathbf{x}_{1:N})p(\mathbf{x}_{1:N}|\mathbf{\Lambda})p(\mathbf{\Lambda})}{q_\phi(\mathbf{x}_{1:N}|\mathbf{y}_{1:N})q_\lambda(\mathbf{\Lambda})} \right] \\
&\qquad \times q_\nu(\mathbf{u}_{1:N}|\mathbf{x}_{1:N}, \mathbf{\Lambda})q_\lambda(\mathbf{\Lambda})q_\phi(\mathbf{x}_{1:N}|\mathbf{y}_{1:N}) \mathrm{d}\mathbf{x}_{1:N} \mathrm{d}\mathbf{u}_{1:N} \mathrm{d}\mathbf{\Lambda} \\
&= \int \log \left[\frac{p_\theta(\mathbf{y}_{1:N}|\mathbf{x}_{1:N})}{q_\phi(\mathbf{x}_{1:N}|\mathbf{y}_{1:N})} \right] q_\phi(\mathbf{x}_{1:N}|\mathbf{y}_{1:N}) \mathrm{d}\mathbf{x}_{1:N} \\
&\qquad + \int \left[\log p(\mathbf{x}_{1:N}|\mathbf{\Lambda}) + \log \frac{p(\mathbf{\Lambda})}{q_\lambda(\mathbf{\Lambda})} \right] q_\lambda(\mathbf{\Lambda})q_\phi(\mathbf{x}_{1:N}|\mathbf{y}_{1:N}) \mathrm{d}\mathbf{x}_{1:N} \mathrm{d}\mathbf{\Lambda} \\
&= \mathbb{E}_{q_\phi} \left[\log \frac{p_\theta(\mathbf{y}_{1:N}|\mathbf{x}_{1:N})}{q_\phi(\mathbf{x}_{1:N}|\mathbf{y}_{1:N})} \right] + \mathbb{E}_{q_\phi} \left[\mathbb{E}_{q_\lambda} \left[\log p(\mathbf{x}_{1:N}|\mathbf{\Lambda}) + \log \frac{p(\mathbf{\Lambda})}{q_\lambda(\mathbf{\Lambda})} \right] \right],
\end{aligned}$$

Using a single MC sample from $q_\phi(\mathbf{x}_{1:N}|\mathbf{y}_{1:N})$ to approximate the expectation, we can write the loss function as:

$$\mathcal{F}(\theta, \phi, \lambda) = \log \frac{p_\theta(\mathbf{y}_{1:N}|\mathbf{x}_{1:N})}{q_\phi(\mathbf{x}_{1:N}|\mathbf{y}_{1:N})} + \mathbb{E}_{q_\lambda} \left[\log p(\mathbf{x}_{1:N}|\mathbf{\Lambda}) \right] - \mathcal{KL}(q_\lambda(\mathbf{\Lambda}), p(\mathbf{\Lambda})).$$

We can sample $q_\lambda(\mathbf{\Lambda})$ to approximate the expectation of $\log p(\mathbf{x}_{1:N}|\mathbf{\Lambda})$. Note that this requires the reparameterisation trick that is used for also used when sampling $\mathbf{x}_{1:N}$. This allows for backpropagation of errors through the sampling step. The KL-divergence can be calculated analytically for the case of Gaussian prior and posterior:

$$\mathcal{F}(\theta, \phi, \lambda) = \log \frac{p_\theta(\mathbf{y}_{1:N}|\mathbf{x}_{1:N})}{q_\phi(\mathbf{x}_{1:N}|\mathbf{y}_{1:N})} + \frac{1}{M} \sum_{i=1}^{M} \left[\log p(\mathbf{x}_{1:N}|\mathbf{\Lambda}^{(i)}) \right] - \mathcal{KL}(q_\lambda(\mathbf{\Lambda}), p(\mathbf{\Lambda})),$$

and approximated via Monte-Carlo otherwise

$$\mathcal{F}(\theta, \phi, \lambda) = \log \frac{p_\theta(\mathbf{y}_{1:N}|\mathbf{x}_{1:N})}{q_\phi(\mathbf{x}_{1:N}|\mathbf{y}_{1:N})} + \frac{1}{M} \sum_{i=1}^{M} \left[\log p(\mathbf{x}_{1:N}|\mathbf{\Lambda}^{(i)}) + \log p(\mathbf{\Lambda}^{(i)}) - \log q_\lambda(\mathbf{\Lambda}^{(i)}) \right].$$

C Further Details on the Dynamic Model

In this work we take the latent dynamical model to be a stochastic ODE or PDE. For an ODE this follows from a standard SDE (Särkkä & Solin, 2019; Øksendal, 2003), given by

$$\mathrm{d}\mathbf{u} = f_\Lambda(\mathbf{u}, t; \Lambda)\mathrm{d}t + \mathbf{L}(t)\mathrm{d}\mathbf{W}(t),$$

where $\mathbf{u} := \mathbf{u}(t) \in \mathbb{R}^{n_u}$, $t \in [0, T]$, $f_\Lambda : \mathbb{R}^{n_u} \times [0, T] \to \mathbb{R}^{n_u}$, $\mathbf{L} : [0, T] \to \mathbb{R}^{n_u \times n_u}$. The noise process $\mathbf{W}(t)$ is a standard vector Brownian motion. The diffusion term $\mathbf{L}(t)$ can be used to describe any *a priori* correlation in the error process dimensions. As stated in the main text, this error process is taken to represent possibly misspecified/unknown physics, which may have been omitted when specifying the model. Discretisation with an explicit Euler-Maruyama scheme (Kloeden & Platen, 1992) gives,

$$\mathbf{u}_n = \mathbf{u}_{n-1} + \Delta_t f_{n-1}(\mathbf{u}_{n-1}; \Lambda) + \mathbf{L}_{n-1}\Delta\mathbf{W}_{n-1}, \quad \Delta\mathbf{W}_{n-1} \sim \mathcal{N}(\mathbf{0}, \Delta_t \mathbf{I}),$$

where $\mathbf{u}_n := \mathbf{u}(n\Delta_t)$, $f_n(\cdot; \Lambda) = f_\Lambda(\cdot, n\Delta_t)$, and so on. This gives a transition density

$$p(\mathbf{u}_n|\mathbf{u}_{n-1}, \Lambda) = \mathcal{N}(\mathbf{u}_{n-1} + \Delta_t f_{n-1}(\mathbf{u}_{n-1}; \Lambda), \Delta_t \mathbf{L}_{n-1}\mathbf{L}_{n-1}^\top),$$

defining a Markov model on the now discretised state vector $\mathbf{u}_n$. To align with the notation introduced in the main text, this gives:

$$p(\mathbf{u}_n|\mathbf{u}_{n-1}, \Lambda) = \mathcal{N}(\mathcal{M}(\mathbf{u}_{n-1}), \mathbf{Q}),$$
$$\mathcal{M}(\mathbf{u}_{n-1}) := \mathbf{u}_{n-1} + \Delta_t f_{n-1}(\mathbf{u}_{n-1}; \Lambda), \quad \mathbf{Q} := \Delta_t \mathbf{L}_{n-1}\mathbf{L}_{n-1}^\top.$$

Due to the structure of the STATFEM discretisation, the fully-discretised underlying model is of the same mathematical form as this ODE case. The difference lies in the dynamics being defined from either a PDE or ODE system. In common cases, a lower dimensional state vector, $\mathbf{u}_n$, typically results for the ODE case in comparison to the PDE case. For the PDE case, entries of the state vector $\mathbf{u}_n$ are coefficients of the finite element basis functions.

For the PDE case, the derivation is similar, with an additional step pre-time-discretisation to spatially discretise the system. This yields a method-of-lines approach (Schiesser, 1991). As in the main text, we consider a generic nonlinear PDE system of the form

$$\partial_t u + L_\Lambda u + F_\Lambda(u) = f + \dot{\xi}, \quad \dot{\xi} \sim \mathcal{GP}(0, \delta(t - t') \cdot k(\mathbf{s}, \mathbf{s}')), \tag{16}$$

where $u := u(\mathbf{s}, t)$, $\xi := \xi(\mathbf{s}, t)$, $f := f(\mathbf{s})$, $\mathbf{s} \in \Omega \subset \mathbb{R}^d$, and $t \in [0, T]$. The operators L_Λ and $F_\Lambda(\cdot)$ are linear and nonlinear differential operators, respectively. The process $\dot{\xi}$ is the derivative of a function-valued Wiener process, whose increments are given by a Gaussian process with the covariance kernel $k(\cdot, \cdot)$. In our examples, we use the squared-exponential covariance function (Williams & Rasmussen, 2006)

$$k(\mathbf{s}, \mathbf{s}') = \rho^2 \exp\left(-\frac{\|\mathbf{s} - \mathbf{s}'\|_2^2}{2\ell^2}\right).$$

Hyperparameters $\{\rho, \ell\}$ are always assumed to be known, being set *a priori*. Further work investigating inference of these hyperparameters is of interest.

As stated in the main text we discretise the linear time-evolving PDE following the STATFEM as in Duffin et al. (2021), for which we refer to for the full details of this approach. In brief, we discretise spatially with finite elements (see, e.g., Brenner & Scott (2008); Thomée (2006), for standard references) then temporally via finite differences. We first multiply equation 16 with a sufficiently smooth test function $v \in V$, where V is an appropriate function space (e.g. the $H_0^1(\Omega)$ Sobolev space (Evans, 2010)) and integrate over the domain Ω to give the weak form (Brenner & Scott, 2008)

$$\langle \partial_t u, v\rangle + \mathcal{A}_\Lambda(u, v) + \langle F_\Lambda(u), v\rangle = \langle f, v\rangle + \langle \dot{\xi}, v\rangle, \quad \forall v \in V.$$

Recall that $\mathcal{A}_\Lambda(\cdot, \cdot)$ is the bilinear form generated from the linear operator L_Λ, and

$$\langle f, g\rangle = \int_\Omega f(\mathbf{s})g(\mathbf{s})\,\mathrm{d}\mathbf{s},$$

the $L^2(\Omega)$ inner product.

Next we introduce a discrete approximation to the domain, $\Omega_h \subseteq \Omega$, having vertices $\{\mathbf{s}_j\}_{j=1}^{n_h}$. This is parameterised by h which indicates the degree of mesh-refinement. We now introduce a finite-dimensional set of polynomial basis functions $\{\phi_j(\mathbf{s})\}_{j=1}^{n_u}$, such that $\phi_i(\mathbf{s}_j) = \delta_{ij}$. In this work these are exclusively the $C^0(\Omega)$ linear polynomial "hat" basis functions. This gives the finite-dimensional function space $V_h = \mathrm{span}\{\phi_j(\mathbf{s})\}_{j=1}^{n_u}$, which is the space we look for solutions

in. Next, we rewrite u and v in terms of these basis functions: $u_h(\mathbf{s},t) = \sum_{j=1}^{n_u} u_j(t)\phi_j(\mathbf{s})$ and $v_h(\mathbf{s},t) = \sum_{j=1}^{n_u} v_j(t)\phi_j(\mathbf{s})$. As the weak form must hold for all $v_h \in V_h$, this is equivalent to holding for all ϕ_j. Thus, the weak form can now be rewritten in terms of this set of basis functions

$$\langle \partial_t u_h, \phi_j \rangle + \mathcal{A}_\Lambda(u_h, \phi_j) + \langle F_\Lambda(u_h), \phi_j \rangle = \langle f, \phi_j \rangle + \langle \dot{\xi}, \phi_j \rangle, \quad j = 1, \ldots, n_u.$$

Note that, in general, u_h and v_h do not necessarily have to be defined on the same function space, but as we use the linear basis functions in this work we stick with this here.

As stated in the main text, this is an SDE over the FEM coefficients $\mathbf{u}(t) = (u_1(t), \ldots, u_{n_u}(t))^\top$, given by

$$\mathbf{M}\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} + \mathbf{A}\mathbf{u} + \mathcal{F}(\mathbf{u}) = \mathbf{b} + \dot{\boldsymbol{\xi}}, \quad \dot{\boldsymbol{\xi}}(t) \sim \mathcal{N}(\mathbf{0}, \delta(t-t') \cdot \mathbf{G})$$

where $\mathbf{M}_{ij} = \langle \phi_i, \phi_j \rangle$, $\mathbf{A}_{ij} = \mathcal{A}_\Lambda(\phi_i, \phi_j)$, $\mathcal{F}(\mathbf{u})_j = \langle F_\Lambda(u_h), \phi_j \rangle$, $\mathbf{b}_j = \langle f, \phi_j \rangle$, and $\mathbf{G}_{ij} = \langle \phi_i, \langle k(\cdot,\cdot), \phi_j \rangle \rangle$. Letting $\mathbf{G} = \mathbf{L}\mathbf{L}^\top$ we can then write this in the familiar notation as above

$$\mathbf{M}\mathrm{d}\mathbf{u} + \mathbf{A}\mathbf{u}\mathrm{d}t + \mathcal{F}(\mathbf{u})\mathrm{d}t = \mathbf{b}\mathrm{d}t + \mathbf{L}\,\mathrm{d}\mathbf{W}(t),$$

from which an Euler-Maruyama time discretisation gives

$$\mathbf{u}_n = (\mathbf{I} - \Delta_t \mathbf{M}^{-1}\mathbf{A})\mathbf{u}_{n-1} - \Delta_t \mathbf{M}^{-1}\mathcal{F}(\mathbf{u}_{n-1}) + \Delta_t \mathbf{M}^{-1}\mathbf{b} + \mathbf{M}^{-1}\mathbf{L}\Delta\mathbf{W}_{n-1},$$

where $\Delta\mathbf{W}_{n-1} \sim \mathcal{N}(0, \Delta_t \mathbf{I})$, eventually defining a transition model of the form

$$p_\Lambda(\mathbf{u}_n|\mathbf{u}_{n-1}) = \mathcal{N}\left((\mathbf{I} - \Delta_t \mathbf{M}^{-1}\mathbf{A})\mathbf{u}_{n-1} - \Delta_t \mathbf{M}^{-1}\mathcal{F}(\mathbf{u}^{n-1}) + \Delta_t \mathbf{M}^{-1}\mathbf{b}, \Delta_t \mathbf{M}^{-1}\mathbf{G}\mathbf{M}^{-\top}\right). \tag{17}$$

Note that also that the STATFEM methodology also allows for implicit discretisations which may be desirable for time-integrator stability. The transition equations for these approaches can be written out in closed form, yet although they give Markovian transition models, the resultant transition densities $p(\mathbf{u}_n|\mathbf{u}_{n-1}, \boldsymbol{\Lambda})$ are not necessarily Gaussian due to the nonlinear dynamics being applied to the current state $\mathbf{u}_n$. Letting $\mathbf{e}_{n-1} = \mathbf{L}\Delta\mathbf{W}_{n-1} \sim \mathcal{N}(\mathbf{0}, \Delta_t \mathbf{G})$, then the implicit Euler is

$$\mathbf{M}\left(\mathbf{u}_n - \mathbf{u}_{n-1}\right) + \Delta_t \mathbf{A}\mathbf{u}_n + \Delta_t \mathcal{F}(\mathbf{u}_n) + \Delta_t \mathbf{b} = \mathbf{e}_{n-1}, \tag{18}$$

and the Crank-Nicolson is

$$\mathbf{M}\left(\mathbf{u}_n - \mathbf{u}_{n-1}\right) + \Delta_t \mathbf{A}\mathbf{u}_{n-1/2} + \Delta_t \mathcal{F}(\mathbf{u}_{n-1/2}) + \Delta_t \mathbf{b} = \mathbf{e}_{n-1}, \tag{19}$$

where $\mathbf{u}_{n-1/2} = \left(\mathbf{u}_n + \mathbf{u}_{n-1}\right)/2$. Furthermore, to compute the marginal measure $p(\mathbf{u}_n|\boldsymbol{\Lambda})$ this also requires integrating over the previous solution $\mathbf{u}_{n-1}$; again due to nonlinear dynamics this will not necessarily be Gaussian.

In each of these cases, therefore, the transition equation is

$$\mathcal{M}(\mathbf{u}_n, \mathbf{u}_{n-1}) = \mathbf{e}_{n-1},$$

where we take, for the implicit Euler,

$$\mathcal{M}(\mathbf{u}_n, \mathbf{u}_{n-1}) = \mathbf{M}\left(\mathbf{u}_n - \mathbf{u}_{n-1}\right) + \Delta_t \mathbf{A}\mathbf{u}_n + \Delta_t \mathcal{F}(\mathbf{u}_n) + \Delta_t \mathbf{b}$$

and, for the Crank-Nicolson,

$$\mathcal{M}(\mathbf{u}_n, \mathbf{u}_{n-1}) = \mathbf{M}\left(\mathbf{u}_n - \mathbf{u}_{n-1}\right) + \Delta_t \mathbf{A}\mathbf{u}_{n-1/2} + \Delta_t \mathcal{F}(\mathbf{u}_{n-1/2}) + \Delta_t \mathbf{b}.$$

In practice due to conservative properties of the Crank-Nicolson discretisation we use this for all our models in this work.

Discretised solutions $\mathbf{u}_n$ are mapped at time n to "pseudo-observations" via the observation process

$$\mathbf{x}_n = \mathbf{H}\mathbf{u}_n + \mathbf{r}_n, \quad \mathbf{r}_n \sim \mathcal{N}(\mathbf{0}, \mathbf{R}).$$

This observation process has the density $p_\nu(\mathbf{x}_n|\mathbf{u}_n)$ where $\nu = \{\mathbf{H}, \mathbf{R}\}$. As stated in the main text, the pseudo-observation operator $\mathbf{H}$ and observational covariance $\mathbf{R}$ are assumed known in this work. We typically use a diagonal covariance, setting $\mathbf{R} = \sigma^2\mathbf{I}$. In the PDE case, for a given STATFEM discretisation as above, these pseudo-observations are assumed to be taken on a user-specified grid, given by $\mathbf{x}_{\mathrm{obs}} \in \mathbb{R}^{n_x}$. The pseudo-observation operator thus acts as an interpolant, such that

$$\mathbf{H}\mathbf{u}_n = \left[u_h(\mathbf{x}^1_{\mathrm{obs}}, n\Delta_t), u_h(\mathbf{x}^2_{\mathrm{obs}}, n\Delta_t), \ldots, u_h(\mathbf{x}^{n_x}_{\mathrm{obs}}, n\Delta_t)\right]^\top.$$

For the ODE case, we have worked with observation operators that extract relevant entries from the state vector. The pseudo-observations are mapped to high-dimensional observed data through a possibly nonlinear observation model which has the probability density $p_\theta(\mathbf{y}_n|\mathbf{x}_n)$. Recall that in this, θ are neural network parameters. This defines the decoding component of our model (see Figure 1).

Nonlinear Filtering for Latent State Estimation. To perform state inference given a set of pseudo-observations we use the ExKF. The ExKF constructs an approximate Gaussian posterior distribution via linearising about the nonlinear model $\mathcal{M}(\cdot)$. The action of the nonlinear $\mathcal{M}(\cdot)$ is approximated via tangent linear approximation. We will derive our filter in the general context of a nonlinear Gaussian SSM given by

$$\begin{aligned} \text{Transition:} \quad & \mathcal{M}(\mathbf{u}_n, \mathbf{u}_{n-1}) = \mathbf{e}_{n-1}, \quad \mathbf{e}_n \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}), \\ \text{Observation:} \quad & \mathbf{x}_n = \mathbf{H}\mathbf{u}_n + \mathbf{r}_n, \quad \mathbf{r}_n \sim \mathcal{N}(\mathbf{0}, \mathbf{R}). \end{aligned}$$

This allows for the use of implicit time-integrators and subsumes the derivation for the explicit case. We assume that at the previous timestep an approximate Gaussian posterior has been obtained, $p(\mathbf{u}_{n-1}|\mathbf{x}_{1:n-1}, \mathbf{\Lambda}) = \mathcal{N}(\mathbf{m}_{n-1}, \mathbf{C}_{n-1})$. For each n the ExKF thus proceeds as:

1. Prediction step. Solve $\mathcal{M}(\hat{\mathbf{m}}_n, \mathbf{m}_{n-1}) = 0$ for $\hat{\mathbf{m}}_n$. Calculate the tangent linear covariance update:
$$\hat{\mathbf{C}}_n = \mathbf{J}_n^{-1}\left(\mathbf{J}_{n-1}\mathbf{C}_{n-1}\mathbf{J}_{n-1}^\top + \mathbf{Q}\right)\mathbf{J}_n^{-\top},$$
where $\mathbf{J}_n = \partial\mathcal{M}/\partial\mathbf{u}_n|_{\hat{\mathbf{m}}_n, \mathbf{m}_{n-1}}$ and $\mathbf{J}_{n-1} = \partial\mathcal{M}/\partial\mathbf{u}_{n-1}|_{\hat{\mathbf{m}}_n, \mathbf{m}_{n-1}}$.
This gives $p(\mathbf{u}_n|\mathbf{x}_{1:n-1}, \mathbf{\Lambda}) = \mathcal{N}(\hat{\mathbf{m}}_n, \hat{\mathbf{C}}_n)$.
2. Update step. Compute the posterior mean $\mathbf{m}_n$ and covariance $\mathbf{C}_n$:
$$\begin{aligned} \mathbf{m}_n &= \hat{\mathbf{m}}_n + \hat{\mathbf{C}}_n\mathbf{H}^T(\mathbf{H}\hat{\mathbf{C}}_n\mathbf{H}^T + \mathbf{R})^{-1}(\mathbf{y}_n - \mathbf{H}\hat{\mathbf{m}}_n), \\ \mathbf{C}_n &= \hat{\mathbf{C}}_n - \hat{\mathbf{C}}_n\mathbf{H}^T(\mathbf{H}\hat{\mathbf{C}}_n\mathbf{H}^T + \mathbf{R})^{-1}\mathbf{H}\hat{\mathbf{C}}_n. \end{aligned}$$
This gives $p(\mathbf{u}_n|\mathbf{x}_{1:n}, \mathbf{\Lambda}) = \mathcal{N}(\mathbf{m}_n, \mathbf{C}_n)$.

The log-marginal likelihood can be calculated recursively, with each term of the log-likelihood computed after each prediction step:

$$\begin{aligned} \log p(\mathbf{x}_{1:N}|\mathbf{\Lambda}) &= \sum_{n=2}^{N} \log p(\mathbf{x}_n|\mathbf{x}_{1:n-1}, \mathbf{\Lambda}), \\ p(\mathbf{x}_n|\mathbf{x}_{1:n-1}, \mathbf{\Lambda}) &= \mathcal{N}(\mathbf{H}\hat{\mathbf{m}}_n, \mathbf{H}\hat{\mathbf{C}}_n\mathbf{H}^T + \mathbf{R}). \end{aligned}$$

Note that although we focus on the ExKF other nonlinear filters could be used; two popular alternatives are the ensemble Kalman filter (Evensen, 2003), or, the particle filter (Doucet et al., 2000). For a linear dynamical model, such as the advection equation considered in Section 5.2, the ExKF reduces to the standard Kalman filter (Kalman, 1960).

As mentioned in the main text we can marginalise over the uncertainty in the encoder, via a Monte Carlo approximation:

$$p(\mathbf{u}_n|\mathbf{y}_{1:n}, \mathbf{\Lambda}) \approx \int p(\mathbf{u}_n|\mathbf{x}_{1:n}, \mathbf{\Lambda}) q_\phi(\mathbf{x}_{1:n}|\mathbf{y}_{1:n})\, \mathrm{d}\mathbf{x}_{1:n} \tag{20}$$

$$\approx \frac{1}{M}\sum_{i=1}^{M} p(\mathbf{u}_n|\mathbf{x}_{1:n}^{(i)}, \mathbf{\Lambda}), \quad \mathbf{x}_{1:n}^{(i)} \sim q_\phi(\cdot|\mathbf{y}_{1:n}). \tag{21}$$

The intractable integral is approximated using samples from the encoder, which provides an approximate posterior in the form of a mixture of Gaussians distribution, where each $p(\mathbf{u}_n|\mathbf{x}_{1:n}^{(i)}, \mathbf{\Lambda}) = \mathcal{N}(\mathbf{m}_n^{(i)}, \mathbf{C}_n^{(i)})$. A similar marginalisation procedure can proceed over the parameters

$$p(\mathbf{u}_n|\mathbf{y}_{1:n}) \approx \int p(\mathbf{u}_n|\mathbf{x}_{1:n}, \mathbf{\Lambda}) q_\lambda(\mathbf{\Lambda}) q_\phi(\mathbf{x}_{1:n}|\mathbf{y}_{1:n})\, \mathrm{d}\mathbf{\Lambda}\mathrm{d}\mathbf{x}_{1:n} \tag{22}$$

$$\approx \frac{1}{M_{\mathbf{x}} M_{\mathbf{\Lambda}}} \sum_{i=1}^{M_{\mathbf{x}}} \sum_{j=1}^{M_{\mathbf{\Lambda}}} p(\mathbf{u}_n|\mathbf{x}_{1:n}^{(i)}, \mathbf{\Lambda}^{(j)}), \quad \mathbf{x}_{1:n}^{(i)} \sim q_\phi(\cdot|\mathbf{y}_{1:n}),\ \mathbf{\Lambda}^{(j)} \sim q_\lambda(\cdot). \tag{23}$$