UNIVERSITY OF AMSTERDAM

UvA-DARE (Digital Academic Repository)

Self-Supervised Inference in State-Space Models

Ruhe, D. ; Forré, P.

DOI
10.48550/arXiv.2107.13349
Publication date
2021
Document Version
Submitted manuscript
License
Unspecified

Link to publication

Citation for published version (APA):
Ruhe, D., & Forré, P. (2021). *Self-Supervised Inference in State-Space Models*. (v1 ed.) ArXiv. https://doi.org/10.48550/arXiv.2107.13349

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, P.O. Box 19185, 1000 GD Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (https://dare.uva.nl)

Download date:15 Sep 2026

arXiv:2107.13349v1 [cs.LG] 28 Jul 2021

Self-Supervised Hybrid Inference in State-Space Models

David Ruhe
AI4Science Lab, AMLab
Informatics Institute, Anton Pannekoek Institute
University of Amsterdam
The Netherlands
david.ruhe@gmail.com

Patrick Forré
AI4Science Lab, AMLab
Informatics Institute
University of Amsterdam
The Netherlands
p.d.forre@uva.nl

Abstract

We perform approximate inference in state-space models that allow for nonlinear higher-order Markov chains in latent space. The conditional independencies of the generative model enable us to parameterize only an inference model, which learns to estimate clean states in a self-supervised manner using maximum likelihood. First, we propose a recurrent method that is trained directly on noisy observations. Afterward, we cast the model such that the optimization problem leads to an update scheme that backpropagates through a recursion similar to the classical Kalman filter and smoother. In scientific applications, domain knowledge can give a linear approximation of the latent transition maps. We can easily incorporate this knowledge into our model, leading to a hybrid inference approach. In contrast to other methods, experiments show that the hybrid method makes the inferred latent states physically more interpretable and accurate, especially in low-data regimes. Furthermore, we do not rely on an additional parameterization of the generative model or supervision via uncorrupted observations or ground truth latent states. Despite our model's simplicity, we obtain competitive results on the chaotic Lorenz system compared to a fully supervised approach and outperform a method based on variational inference.

1 Introduction

Many sequential processes in industry and research involve noisy measurements that describe latent dynamics. Given these corrupted observations, we are interested in obtaining optimal clean state estimates. Radio astronomy provides a use-case, which we use exemplary throughout this introduction. We want to obtain a noise-free image of an extremely distant object. The data that a radio interferometer receives is noisy and sparse [Thompson et al., 2017]. Often the mechanism that induces the sparsity is known, and there are accurate estimates of the noise of the measuring device, here the interferometer [Thompson et al., 2017]. It is also known that by collecting data over time, a less sparse and less noisy measurement can be obtained (e.g., Bouman et al. [2016]), commonly referred to as Earth rotation synthesis [Thompson et al., 2017]. However, performing inference instead could provide better images, especially in the real-time case (e.g., [Kuiack et al., 2019, Ellingson et al., 2013, Ruhe et al., 2021].

State-space models are proven to be a powerful tool for representing such noise-afflicted latent dynamics. Approaches to perform posterior inference in state-space models include the celebrated Kalman filter [Kalman, 1960] and Rauch-Tung-Striebel (RTS) smoother [Rauch et al., 1965]. They are proven to be optimal in the linear Gaussian case. However, the transitions in radio astronomy involve a non-linear process that is influenced by e.g. satellites, airplanes, meteors, space weather, and atmospheric turbulence. Therefore, work has been put in finding optimal solutions to perform

Preprint. Under review.

Figure 1: Best viewed on screen. Qualitative results of our work. To the noisy measurements (1st from left) we apply an extended Kalman smoother (2nd). From the noisy measurements we learn a recurrent inference model that does slightly better (3rd). Our hybrid model combines expert knowledge with inference (4th), yielding the best result. Ground truth provided for comparison (5th).

inference in non-linear state-space models. Early methods include the Extended Kalman filter [Ljung, 1979] and the Unscented Kalman filter [Wan and Van Der Merwe, 2000] that use linearization techniques to approximate the transitions and emissions. Other (newer) approaches include Murphy and Russell [2001], Liu and Chen [1998], Kitagawa [1996], Arasaratnam and Haykin [2009], Briers et al. [2010]. However, these still rely on Monte Carlo sampling, linearization, or other techniques that approximate higher-dimensional weighted integrals. Therefore, they can be numerically unstable or inaccurate. Moreover, the parameters of these models are mostly based on pre-defined knowledge or they are approximately inferred. Methods that utilize the recent success of deep learning include Krishnan et al. [2015], Archer et al. [2015], Chung et al. [2015], Krishnan et al. [2017], Rangapuram et al. [2018], Lin et al. [2018], Becker et al. [2019], Karl et al. [2017], Johnson et al. [2016], Bayer and Osendorfer [2014], Haarnoja et al. [2016], Satorras et al. [2019]. To these approaches, there usually are several limitations: (1) it is assumed that a training dataset containing ground truth states or measurements is available, (2) full black-box variational inference is directly applied, discarding the requirement that the latent states should represent something physical, (3) inference using a lower bound can be hard to optimize, and convergence is hard to determine (4) knowledge about the emission distribution (i.e., the measurement device) is not included, (5) prior knowledge about the transition function is not included, (6) there might not be enough data available for regular deep learning approaches, (7) no flexible models are directly learned from the data or (8) the transition probabilities are assumed to be first-order Markov. If one or more of these are the case, there can be limited practical value.

In this paper, we present our approach to addressing all these challenges. Our contributions can be summarized as follows:

1. We perform approximate inference in state-space models that allow for nonlinear higher-order Markov chains in latent space under mild assumptions on the generative process.
2. Our method can learn flexible functions using maximum likelihood in a self-supervised manner. That is, ground truth values of states and measurements are not assumed to be available for training.
3. Our approach can but does not need to utilize knowledge about the emission distribution (e.g., a known selection matrix and noise estimates in radio astronomy).
4. A recursive method that is akin to the Kalman filter and smoother updates and can be combined with prior knowledge about the transition probabilities into a hybrid model (e.g., the rotation of the Earth as is done in rotation synthesis [Thompson et al., 2017]). This allows for better applicability in low data regimes and incentivizes the model to provide more interpretable estimates of the latent states.

Among others, we verify our method on the same experiments[1] as recent supervised work by Satorras et al. [2019], and obtain similar results and outperform a variational approach [Krishnan et al., 2017].

The paper is structured as follows. In section 2 we describe the assumptions that we make about our generative model. In section 3 we show how we parameterize our inference model for efficient fitting and inference. In section 4 we introduce how the model can be used to obtain a Bayesian posterior estimate of the state. In section 5 we set up our recurrent inference model. In section 6 we include

[1] Available at: `https://github.com/DavidRuhe/self-supervised-hybrid-inference`

expert knowledge into a recursive hybrid model that includes Kalman filter and smoother updates. In section 7 we outline our experiments and results. In section 8 we discuss and conclude.

2 Generative Model Assumptions

In this section, we explicitly state the model's required generative process assumptions. First, we assume that we can measure (at least) one run of (noise-afflicted) sequential data $\mathbf{y}_{0:K} := (\mathbf{y}_0, \ldots, \mathbf{y}_K)$, where each $\mathbf{y}_k \in \mathbb{R}^M$, $k = 0, \ldots, K$. We abbreviate: $\mathbf{y}_{l:k} := (\mathbf{y}_l, \ldots, \mathbf{y}_k)$ and $\mathbf{y}_{<k} := \mathbf{y}_{0:k-1}$ and $\mathbf{y}_{\leq k} := \mathbf{y}_{0:k}$ and $\mathbf{y}_{\neg k} := (\mathbf{y}_{0:k-1}, \mathbf{y}_{k+1:K})$. In our example, the $\mathbf{y}_k$ would be noisy radio signals from some area in space measured over time points $k = 0, \ldots, K$. We then assume that $\mathbf{y}_{0:K}$ is the result of some possibly non-linear probabilistic latent dynamics, i.e. of a distribution $p(\mathbf{x}_{0:K})$, whose variables are given by $\mathbf{x}_{0:K} := (\mathbf{x}_0, \ldots, \mathbf{x}_K)$ with $\mathbf{x}_k \in \mathbb{R}^N$. Each $\mathbf{y}_k$ is assumed to be drawn from some shared noisy emission probability $p(\mathbf{y}_k \mid \mathbf{x}_k)$. The joint probability is then assumed to factorize as:

$$p(\mathbf{y}_{0:K}, \mathbf{x}_{0:K}) = \prod_{k=0}^{K} p(\mathbf{y}_k \mid \mathbf{x}_k) \cdot p(\mathbf{x}_{0:K}). \quad (1)$$

Note that this factorization encodes several conditional independencies like $\mathbf{y}_k \perp\!\!\!\perp (\mathbf{y}_{\neg k}, \mathbf{x}_{\neg k}) \mid \mathbf{x}_k$. Typical models that follow these assumptions are linear dynamical systems, hidden Markov models, but also nonlinear state-space models with higher-order Markov chains in latent space, like depicted in fig. 2 and the supplementary material.

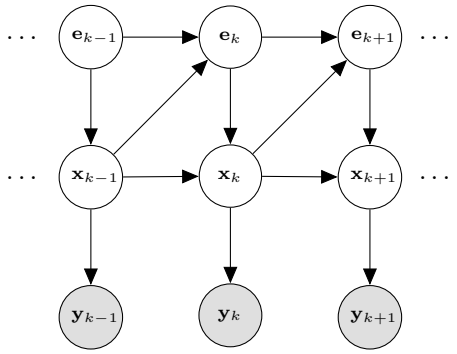

Figure 2: State-space model with deeper latent structure.

In contrast to other approaches (e.g., Krishnan et al. [2015, 2017]) where one tries to model the latent dynamics with transition probabilities $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$ and possibly non-linear emission probabilities $p(\mathbf{y}_k \mid \mathbf{x}_k)$, we go the other way around. We assume that all the non-linear dynamics is captured inside the latent distribution $p(\mathbf{x}_{0:K})$, where we make no further assumption about its factorization, and the emission probabilities are (well-approximated with) a linear Gaussian noise model (also see Ljung [1979], Wan and Van Der Merwe [2000]):

$$p(\mathbf{y}_k \mid \mathbf{x}_k) = \mathcal{N}(\mathbf{y}_k \mid \mathbf{H}\,\mathbf{x}_k, \mathbf{R}), \quad (2)$$

where the matrix $\mathbf{H}$ represents the measurement device and $\mathbf{R}$ is the covariance matrix of the independent additive noise. Note that the Gaussianity of the additive noise is only strictly required for the recursive model (section 6). Returning to the astronomy example of our introduction, $\mathbf{H}$ represents the interferometer that sparsely measures the radio sky under (usually Gaussian) noise [Thompson et al., 2017]. Another frequently found use case is when $\mathbf{H}$ is the identity matrix. That is, the state is fully observed but with noise, in which the inference problem reduces to a denoising task.

To enable a linearized smoothing algorithm in section 6.2 we also need to assume that the conditional mutual information $I(\mathbf{x}_{k-1}; \mathbf{y}_{k:K} \mid \mathbf{x}_k, \mathbf{y}_{0:k-1})$ is small for all $k = 1, \ldots, K$. In other words, we assume that we approximately have the following conditional independences:

$$\mathbf{x}_{k-1} \perp\!\!\!\perp \mathbf{y}_{k:K} \mid (\mathbf{x}_k, \mathbf{y}_{0:k-1}). \quad (3)$$

To explain the motivation for this requirement consider the model in fig. 2. If now the states of $\mathbf{y}_{0:k-1}$ and $\mathbf{x}_k$ are known then the additional information that $\mathbf{y}_{k:K}$ has about the state of the latent variable $\mathbf{x}_{k-1}$ would need to be passed along the unblocked deeper paths like: $\mathbf{y}_{k+1} \leftarrow \mathbf{x}_{k+1} \leftarrow \mathbf{e}_{k+1} \leftarrow \mathbf{e}_k \leftarrow \mathbf{x}_{k-1}$. Then the assumption of small $I(\mathbf{x}_{k-1}; \mathbf{y}_{k:K} \mid \mathbf{x}_k, \mathbf{y}_{0:k-1})$ can be interpreted as that the deeper paths transport less information than the lower direct paths. If we consider all edges to the $\mathbf{x}_k$'s as linear and the edges to the $\mathbf{e}_k$'s as non-linear maps the above could be interpreted as an information theoretic version of that the non-linear correction terms are small compared to the linear parts in the functional relations between the variables. This assumption will only be needed in section 6.2 to make linearized smoothing viable.

Further implicit assumptions about the generative model are imposed via inference model choices later on.

3 Parameterizing and Fitting the Inference Model

In this section, we show how to generally parameterize the inference model in such a way that we easily can fit it to data using maximum likelihood estimation. We also discuss how forecasting works.

Given our noisy measurements $\mathbf{y}_{0:K} = (\mathbf{y}_0, \ldots, \mathbf{y}_K)$ we want to find good estimates for the latent states $\mathbf{x}_{0:K} = (\mathbf{x}_0, \ldots, \mathbf{x}_K)$, which generated the $\mathbf{y}_{0:K}$. For this we want to infer the marginal conditional distributions $p(\mathbf{x}_k \mid \mathbf{y}_{\leq k})$ or $p(\mathbf{x}_k \mid \mathbf{y}_{<k})$ (for forecasting), for an online inference approach (*filtering*); and $p(\mathbf{x}_k \mid \mathbf{y}_{0:K})$ or $p(\mathbf{x}_k \mid \mathbf{y}_{\neg k})$, for a full inference approach (*smoothing*). Let us first consider parameterized filtering. The parameterized smoothing case is treated similarly.

A key insight for efficient inference is the following. Instead of parameterizing the latent transition probabilities of the generative model, an approach that becomes bothersome for non-linear higher-order latent Markov chains, we instead directly parameterize the following conditional probability distributions for the filtering case as:

$$p(\mathbf{x}_k \mid \mathbf{y}_{<k}) = \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{x}}_{k|<k}(\mathbf{y}_{<k}), \hat{\mathbf{P}}_{k|<k}(\mathbf{y}_{<k})\right), \tag{4}$$

with recurrent neural network parameterizations for $\hat{\mathbf{x}}_{k|<k}(\mathbf{y}_{<k})$ and $\hat{\mathbf{P}}_{k|<k}(\mathbf{y}_{<k})$, which we will discuss in section 5 and section 6 in more detail.

We can use the parameterization from eq. (4) and the emission model $p(\mathbf{y}_k \mid \mathbf{x}_k) = \mathcal{N}(\mathbf{y}_k \mid \mathbf{H}\,\mathbf{x}_k, \mathbf{R})$ from eq. (2) together with the factorization from eq. (1) to get an analytic form of the log-likelihood of the data:

$$p(\mathbf{y}_k \mid \mathbf{y}_{<k}) = \int p(\mathbf{y}_k \mid \mathbf{x}_k)\, p(\mathbf{x}_k \mid \mathbf{y}_{<k})\, d\mathbf{x}_k \tag{5}$$

$$= \mathcal{N}\left(\mathbf{y}_k \mid \mathbf{H}\,\hat{\mathbf{x}}_{k|<k}(\mathbf{y}_{<k}), \mathbf{H}\,\hat{\mathbf{P}}_{k|<k}(\mathbf{y}_{<k})\,\mathbf{H}^\top + \mathbf{R}\right), \tag{6}$$

$$\log p(\mathbf{y}_{0:K}) = \sum_{k=0}^{K} \log \mathcal{N}\left(\mathbf{y}_k \mid \mathbf{H}\,\hat{\mathbf{x}}_{k|<k}(\mathbf{y}_{<k}), \mathbf{H}\,\hat{\mathbf{P}}_{k|<k}(\mathbf{y}_{<k})\,\mathbf{H}^\top + \mathbf{R}\right). \tag{7}$$

If we put $\hat{\mathbf{y}}_{k|<k} := \mathbf{H}\,\hat{\mathbf{x}}_{k|<k}(\mathbf{y}_{<k})$ and $\hat{\mathbf{M}}_{k|<k} := \mathbf{H}\,\hat{\mathbf{P}}_{k|<k}(\mathbf{y}_{<k})\,\mathbf{H}^\top + \mathbf{R}$ then the maximum-likelihood objective leads to the following loss function, which we can minimize using gradient descent methods w.r.t. all model parameters:

$$\mathcal{L} := \sum_{k=0}^{K} \left[(\hat{\mathbf{y}}_{k|<k} - \mathbf{y}_k)^\top\, \hat{\mathbf{M}}_{k|<k}^{-1}\, (\hat{\mathbf{y}}_{k|<k} - \mathbf{y}_k) + \log\det \hat{\mathbf{M}}_{k|<k} \right]. \tag{8}$$

Note that each term in the sum above represents a one-step ahead *self-supervised* error term. We thus minimize the prediction residuals $\hat{\mathbf{y}}_{k|<k}(\mathbf{y}_{<k}) - \mathbf{y}_k$ in a norm that is inversely scaled with the above covariance matrix, plus a regularizing determinant term, which prevents the covariance matrix from diverging. The arisen objective function is similar to e.g. Krull et al. [2019], Batson and Royer [2019], Laine et al. [2019] from computer vision literature, but a similar procedure was also used in the causality literature by Schölkopf et al. [2016]. A bias-variance composition (that is similar to the one in e.g. Batson and Royer [2019]) is given in the supplementary material.

For the smoothing case we would, similar to before, parameterize:

$$p(\mathbf{x}_k \mid \mathbf{y}_{\neg k}) = \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{x}}_{k|\neg k}(\mathbf{y}_{\neg k}), \hat{\mathbf{P}}_{k|\neg k}(\mathbf{y}_{\neg k})\right), \tag{9}$$

where here we now use some form of two-sided recurrent neural network for $\hat{\mathbf{x}}_{k|\neg k}(\mathbf{y}_{\neg k})$ and $\hat{\mathbf{P}}_{k|\neg k}(\mathbf{y}_{\neg k})$, see section 5 and section 6. It is worth mentioning for this case the similarities with leave-one-out cross-validation. Further details are given in the supplementary material.

Forecasting After fitting the parameters to the data eq. (4) can then directly be used to do one-step ahead forecasting. By plugging the new value $\hat{\mathbf{x}}_{K+1|K}$ via $\mathbf{y}_{K+1} := \mathbf{H}\,\hat{\mathbf{x}}_{K+1|K}$ back into the recurrent model, and so on, forecasting to arbitrary time steps is possible.

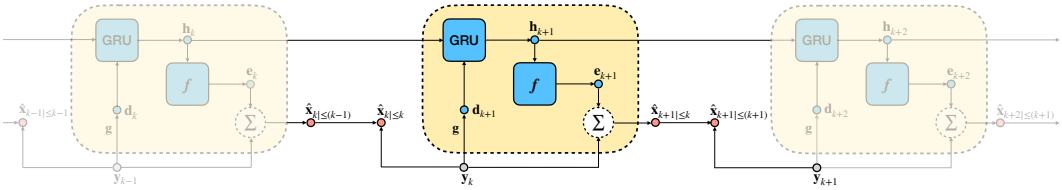

Figure 3: Our recurrent inference model in a filtering setting. The data $\mathbf{y}_k$ is transformed into a stable input $\mathbf{d}_{k+1}$ after which a GRU unit takes it and outputs a new hidden state $\mathbf{h}_{k+1}$. A decoder $\boldsymbol{f}$ outputs a correction $\mathbf{e}_{k+1}$ that is added into a prior mean estimate $\hat{\mathbf{x}}_{k+1|\leq k}$. Using the measurement $\mathbf{y}_{k+1}$ we arrive at posterior mean estimate $\hat{\mathbf{x}}_{k+1|\leq k+1}$.

4 Improving the Inference

After we have seen in eq. (4) how to generally parameterize $p(\mathbf{x}_k \mid \mathbf{y}_{<k})$ and $p(\mathbf{x}_k \mid \mathbf{y}_{\neg k})$, resp., and fit it to data, we now want to improve the inference for $\mathbf{x}_k$ by also including the present data point $\mathbf{y}_k$. By Bayes' rule and the conditional independence $\mathbf{y}_k \perp\!\!\!\perp \mathbf{y}_{\neg k} | \mathbf{x}_k$ from eq. (1) we can derive the following equations for $p(\mathbf{x}_k \mid \mathbf{y}_{\leq k}) = p(\mathbf{x}_k \mid \mathbf{y}_k, \mathbf{y}_{<k})$:

$$p(\mathbf{x}_k \mid \mathbf{y}_k, \mathbf{y}_{<k}) = \frac{p(\mathbf{y}_k \mid \mathbf{x}_k, \mathbf{y}_{<k}) \cdot p(\mathbf{x}_k \mid \mathbf{y}_{<k})}{p(\mathbf{y}_k \mid \mathbf{y}_{<k})} = \frac{p(\mathbf{y}_k \mid \mathbf{x}_k)}{p(\mathbf{y}_k \mid \mathbf{y}_{<k})} \cdot p(\mathbf{x}_k \mid \mathbf{y}_{<k}). \tag{10}$$

Note that the analogous eq. (10) holds for $p(\mathbf{x}_k \mid \mathbf{y}_{0:K}) = p(\mathbf{x}_k \mid \mathbf{y}_k, \mathbf{y}_{\neg k})$ when we replace $\mathbf{y}_{<k}$ with $\mathbf{y}_{\neg k}$ everywhere on the conditioning side.

For the filtering case, our chosen parameterization in eq. (2), $p(\mathbf{y}_k \mid \mathbf{x}_k) = \mathcal{N}(\mathbf{y}_k \mid \mathbf{H}\,\mathbf{x}_k, \mathbf{R})$, and eq. (4), $p(\mathbf{x}_k \mid \mathbf{y}_{<k}) = \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{x}}_{k|<k}(\mathbf{y}_{<k}), \hat{\mathbf{P}}_{k|<k}(\mathbf{y}_{<k})\right)$, together with eq. (10) and the usual matrix formulas now allow us to also get an analytic expression for:

$$p(\mathbf{x}_k \mid \mathbf{y}_{\leq k}) = \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{x}}_{k|\leq k}(\mathbf{y}_{\leq k}), \hat{\mathbf{P}}_{k|\leq k}(\mathbf{y}_{<k})\right), \tag{11}$$

with the following abbreviations:

$$\hat{\mathbf{x}}_{k|\leq k} := \hat{\mathbf{x}}_{k|<k} + \hat{\mathbf{K}}_k\,(\mathbf{y}_k - \mathbf{H}\,\hat{\mathbf{x}}_{k|<k}), \qquad \hat{\mathbf{P}}_{k|\leq k} := \hat{\mathbf{P}}_{k|<k} - \hat{\mathbf{K}}_k\,\mathbf{H}\,\hat{\mathbf{P}}_{k|<k}, \tag{12}$$

where we introduce the *Kalman gain matrix* similar to the classical formulas:

$$\hat{\mathbf{K}}_k := \hat{\mathbf{P}}_{k|<k}\,\mathbf{H}^\top \left(\mathbf{H}\,\hat{\mathbf{P}}_{k|<k}\,\mathbf{H}^\top + \mathbf{R}\right)^{-1}. \tag{13}$$

Note that taking the matrix inverse at this place in eq. (13) is more efficient than in the standard Gaussian formulas (appendix A) if $M \leq N$, which holds for our experiments.

For the parameterized smoothing case the updates are highly similar, except the parameterized estimates $\hat{\mathbf{x}}_{k|k}(\mathbf{y}_{k|\neg k})$ and $\hat{\mathbf{P}}_{k|\neg k}(\mathbf{y}_{\neg k})$ now also include future observations. For more detail the reader is referred to the supplementary material.

5 The Recurrent Model

In eq. (4) we saw how to generally parameterize and fit the conditional probability: $p(\mathbf{x}_k \mid \mathbf{y}_{\neg k}) = \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{x}}_{k|\neg k}(\mathbf{y}_{\neg k}), \hat{\mathbf{P}}_{k|\neg k}(\mathbf{y}_{\neg k})\right)$. Here we want to go into more detail for a specific case of the two-sided recurrent neural networks $\hat{\mathbf{x}}_{k|\neg k}(\mathbf{y}_{\neg k})$ and $\hat{\mathbf{P}}_{k|\neg k}(\mathbf{y}_{\neg k})$. That is, we focus on the smoothing case. Filtering can be done by leaving out everything that depends on future data and by replacing $\mathbf{y}_{\neg k}$ with $\mathbf{y}_{<k}$. We model:

$$\hat{\mathbf{x}}_{k|\neg k} := \mathbf{H}^\top \mathbf{y}_{k-1} + \mathbf{e}_k, \qquad \hat{\mathbf{P}}_{k|\neg k} := \mathbf{L}_k \mathbf{L}_k^\top, \qquad \begin{bmatrix} \mathbf{e}_k \\ \mathbf{L}_k \end{bmatrix} := \boldsymbol{f}(\mathbf{h}_k^{\rightarrow}, \mathbf{h}_k^{\leftarrow}), \tag{14}$$

where $\mathbf{L}_k$ is a lower-triangular matrix with positive diagonal and $\boldsymbol{f}$ is a multi-layer perceptron. The arguments $\mathbf{h}_k^{\rightarrow}$ and $\mathbf{h}_k^{\leftarrow}$ are recursively given by the following equations:

$$\mathbf{h}_k^{\rightarrow} := \mathrm{GRU}(\mathbf{d}_k^{\rightarrow}, \mathbf{h}_{k-1}^{\rightarrow}), \qquad \mathbf{h}_k^{\leftarrow} := \mathrm{GRU}(\mathbf{d}_k^{\leftarrow}, \mathbf{h}_{k+1}^{\leftarrow}), \tag{15}$$

$$\mathbf{d}_k^{\rightarrow} := \mathbf{g}^{\rightarrow}(\mathbf{y}_{\neg k}), \qquad \mathbf{d}_k^{\leftarrow} := \mathbf{g}^{\leftarrow}(\mathbf{y}_{\neg k}), \tag{16}$$

where we employ a Gated Recurrent Unit (GRU) network [Chung et al., 2014]. For numerical stability, we cast the available data into a more scale-invariant form using a pre-processing function $\mathbf{g}(\mathbf{y}_{\neg k})$. In our experiments it suffices to use $\mathbf{g}_k^{\rightarrow}(\mathbf{y}_{\neg k}) = \mathbf{y}_{k-2} - \mathbf{y}_{k-1}$ and $\mathbf{g}_k^{\leftarrow}(\mathbf{y}_{\neg k}) = \mathbf{y}_{k+2} - \mathbf{y}_{k+1}$. Furthermore, we amortize the above functions/probability distributions by weight sharing. An overview is given in fig. 3.

6 The Recursive Hybrid Model

In this section, we introduce a very flexible and convenient parameterization that will neatly fit into the Kalman filter and Kalman smoother recurrence, resp., but at the same time also allows for data-dependent transition and covariance matrices. They can effectively be combined with expert domain knowledge, which we do in our experiments. We will therefore refer to those as *hybrid filter* and *hybrid smoother*.

6.1 Hybrid Filtering

We have seen in eq. (10) how our model assumptions lead to a relation between $p(\mathbf{x}_k \mid \mathbf{y}_{<k})$ and $p(\mathbf{x}_k \mid \mathbf{y}_{\leq k})$. This was used in section 4 to show how the Gaussianity assumptions $p(\mathbf{x}_k \mid \mathbf{y}_{<k})$ conditioned on $\mathbf{y}_{<k}$ propagates to $p(\mathbf{x}_k \mid \mathbf{y}_{\leq k})$ when further conditioned on $\mathbf{y}_k$. In this section we will exploit a simple chain rule that will give us another relation between $p(\mathbf{x}_{k+1} \mid \mathbf{y}_{\leq k})$ and $p(\mathbf{x}_k \mid \mathbf{y}_{\leq k})$, which allows us to integrate this process to the next time step:

$$p(\mathbf{x}_{k+1} \mid \mathbf{y}_{\leq k}) = \int p(\mathbf{x}_{k+1} \mid \mathbf{x}_k, \mathbf{y}_{\leq k})\, p(\mathbf{x}_k \mid \mathbf{y}_{\leq k})\, d\mathbf{x}_k. \tag{17}$$

Note that the right factor is the posterior eq. (10) of the previous time-step, introducing a recursion. To make this work we propose the following advantageous parameterization:

$$p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{<k}) = \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{F}}_{k|<k}\, \mathbf{x}_{k-1} + \hat{\mathbf{e}}_{k|<k}, \hat{\mathbf{Q}}_{k|<k}\right), \tag{18}$$

where $\hat{\mathbf{F}}_{k|<k} := \hat{\mathbf{F}}_{k|<k}(\mathbf{y}_{<k})$ and $\hat{\mathbf{e}}_{k|<k} := \hat{\mathbf{e}}_{k|<k}(\mathbf{y}_{<k})$ and $\hat{\mathbf{Q}}_{k|<k} := \hat{\mathbf{Q}}_{k|<k}(\mathbf{y}_{<k})$ are parameterized with recurrent neural networks, similar to section 5. Equation (18) and eq. (17) then lead to:

$$\hat{\mathbf{P}}_{k|<k} = \hat{\mathbf{F}}_{k|<k}\, \hat{\mathbf{P}}_{k-1|\leq k-1}\, \hat{\mathbf{F}}_{k|<k}^{\top} + \hat{\mathbf{Q}}_{k|<k}, \qquad \hat{\mathbf{x}}_{k|<k} = \hat{\mathbf{F}}_{k|<k}\, \hat{\mathbf{x}}_{k-1|\leq k-1} + \hat{\mathbf{e}}_{k|<k}, \tag{19}$$

which together with eq. (13) and eq. (12) determine all $\hat{\mathbf{x}}_{k|<k}$, $\hat{\mathbf{x}}_{k|\leq k}$, $\hat{\mathbf{P}}_{k|<k}$ and $\hat{\mathbf{P}}_{k|\leq k}$ recursively for all time points when the $\hat{\mathbf{F}}_{k|<k}$, $\hat{\mathbf{e}}_{k|<k}$ and $\hat{\mathbf{Q}}_{k|<k}$ are provided akin to the classical Kalman filter. This approach thus augments the classical Kalman filter with learned transition and covariance matrices. In our experiments, we let $\hat{\mathbf{F}}_{k|<k} := \mathbf{F}_k$ be physics-based transition matrices (independent of $\mathbf{y}_{0:K}$) to replicate the experiments as performed by Satorras et al. [2019]. Since this combines known physics with learned knowledge, we also refer to this as the hybrid model in the experiments section. If needed, we add a ℓ_2-regularizer to the loss function that keeps $\hat{\mathbf{e}}_{k|<k}$ small. By doing so, the model is incentivized to only add a minimal nonlinear correction term to the physics model. A procedure that leads to highly similar updates can be obtained for smoothing, which we detail in the supplementary material.

6.2 Linearized Smoothing

The hybrid filtering from section 6.1 now allows for an additional *linearized smoothing* procedure. The key advantage of this approach is that no additional model has to be trained (which can be costly) to perform smoothing. Several smoothing algorithms stemming from the Kalman filter literature can be applied, such as the RTS smoother [Rauch et al., 1965] and the two-filter smoother [Kitagawa, 1994]. For this to work we need to make the linearization assumption stated in eq. (3).

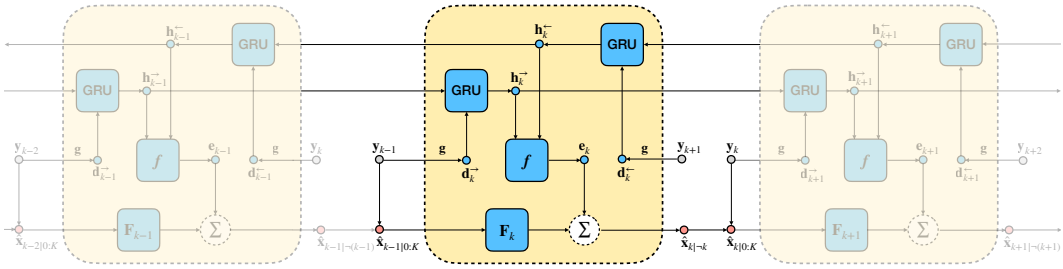

Figure 4: Our recursive hybrid model in a smoothing fashion. Data-points $\mathbf{y}_{k-1}$ and $\mathbf{y}_{k+1}$ are fed into GRU units after pre-processing. Their hidden states are decoded into an additive component $\mathbf{e}_k$. We can thus capture a nonlinear dependency on the observed data. This is combined with a learned or pre-defined linear transition $\mathbf{F}_k$ and previous posterior mean $\hat{\mathbf{x}}_{k-1|0:K}$ into a prior mean estimate $\mathbf{x}_{k|\neg k}$. $\mathbf{y}_{k+1}$ is used to obtain posterior mean $\mathbf{x}_{k+1|0:K}$.

Recall that for smoothing we are interested in the quantity $p(\mathbf{x}_k \mid \mathbf{y}_{0:K})$. We will now show that under the hybrid filter assumptions and eq. (3) we get a Gaussian approximation: $p(\mathbf{x}_k \mid \mathbf{y}_{0:K}) \approx \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{z}}_k, \hat{\mathbf{G}}_k\right)$. We will do backwards induction with $\hat{\mathbf{z}}_K := \hat{\mathbf{x}}_{K|\leq K}$ and $\hat{\mathbf{G}}_K := \hat{\mathbf{P}}_{K|\leq K}$. To propagate this to previous time steps $k-1$ we use the chain rule:

$$p(\mathbf{x}_{k-1} \mid \mathbf{y}_{0:K}) = \int p(\mathbf{x}_{k-1} \mid \mathbf{x}_k, \mathbf{y}_{0:K})\, p(\mathbf{x}_k \mid \mathbf{y}_{0:K})\, d\mathbf{x}_k, \tag{20}$$

where the second term $p(\mathbf{x}_k \mid \mathbf{y}_{0:K}) \approx \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{z}}_k, \hat{\mathbf{G}}_k\right)$ is known by backwards induction and for the first term we make use of the approximate conditional independence $\mathbf{x}_{k-1} \perp\!\!\!\perp \mathbf{y}_{k:K} \mid (\mathbf{x}_k, \mathbf{y}_{0:k-1})$ from eq. (3) to get:

$$p(\mathbf{x}_{k-1} \mid \mathbf{x}_k, \mathbf{y}_{0:K}) = p(\mathbf{x}_{k-1} \mid \mathbf{y}_{k:K}, \mathbf{x}_k, \mathbf{y}_{0:k-1}) \;\approx\; p(\mathbf{x}_{k-1} \mid \mathbf{x}_k, \mathbf{y}_{0:k-1}), \tag{21}$$

where the latter can be shown to be Gaussian by the hybrid filter, which gives:

$$p(\mathbf{x}_{k-1}, \mathbf{x}_k \mid \mathbf{y}_{0:k-1}) = \mathcal{N}\left(\begin{bmatrix}\mathbf{x}_{k-1}\\ \mathbf{x}_k\end{bmatrix} \mid \begin{bmatrix}\hat{\mathbf{x}}_{k-1|\leq k-1}\\ \hat{\mathbf{x}}_{k|<k}\end{bmatrix}, \begin{bmatrix}\hat{\mathbf{P}}_{k-1|\leq k-1} & \hat{\mathbf{P}}_{k-1|\leq k-1}\,\hat{\mathbf{F}}_{k|<k}^\top \\ \hat{\mathbf{F}}_{k|<k}\,\hat{\mathbf{P}}_{k-1|\leq k-1} & \hat{\mathbf{P}}_{k|<k}\end{bmatrix}\right). \tag{22}$$

By use of the usual formulas for Gaussians and the *reverse Kalman gain matrix* $\hat{\mathbf{J}}_{k-1|k}$ we arrive at the following update formulas, $k = K, \ldots, 1$, with $\hat{\mathbf{z}}_K := \hat{\mathbf{x}}_{K|\leq K}$ and $\hat{\mathbf{G}}_K := \hat{\mathbf{P}}_{K|\leq K}$:

$$\hat{\mathbf{J}}_{k-1|k} := \hat{\mathbf{P}}_{k-1|\leq k-1}\,\hat{\mathbf{F}}_{k|<k}^\top\,\hat{\mathbf{P}}_{k|<k}^{-1}, \tag{23}$$

$$\hat{\mathbf{G}}_{k-1} := \hat{\mathbf{P}}_{k-1|\leq k-1} + \hat{\mathbf{J}}_{k-1|k}\left(\hat{\mathbf{P}}_{k|\leq k} - \hat{\mathbf{P}}_{k|<k}\right)\hat{\mathbf{J}}_{k-1|k}^\top, \tag{24}$$

$$\hat{\mathbf{z}}_{k-1} := \hat{\mathbf{x}}_{k-1|\leq k-1} + \hat{\mathbf{J}}_{k-1|k}\,(\hat{\mathbf{z}}_k - \hat{\mathbf{x}}_{k|\leq k}). \tag{25}$$

With these quantities we can do inference for all $k = 0, \ldots, K$ via the approximation:

$$p(\mathbf{x}_k \mid \mathbf{y}_{0:K}) \approx \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{z}}_k, \hat{\mathbf{G}}_k\right). \tag{26}$$

7 Experiments

First, we test and compare our models in a filtering setting to a linear dynamics simulation where the states are not fully observed (sparsely sampled). Then, we test our models on the Lotka-Volterra equations modeling a biological system. Finally, we apply smoothers to a chaotic Lorenz system. For technical details on the setup of the experiments, we refer the reader to the supplementary material.

7.1 Linear Dynamics

In the linear Gaussian case, it is known that the Kalman filter will give the optimal solution. Thus, we can get a lower bound on the test loss. In this toy experiment, we simulate particle tracking under linear dynamics and noisy measurements of the location. We use Newtonian physics equations as prior knowledge. We generate trajectories $\mathcal{T} = \{\mathbf{x}_{0:K}, \mathbf{y}_{0:K}\}$ with $\mathbf{x}_k \in \mathbb{R}^6$ and $\mathbf{y}_k \in \mathbb{R}^2$ according to the differential equations:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -c & 1 \\ 0 & -\tau c & 0 \end{bmatrix} \begin{bmatrix} p \\ v \\ a \end{bmatrix} \tag{27}$$

We simulate a noisy version of this system and sample using rate $\Delta t = 1$ To simulate a model of limited understanding $\mathbf{F}_k$, we use the standard Taylor series of the matrix exponential of $\mathbf{A} \cdot \Delta t$ up to the first order. This is the form we use in the Kalman filter solution and the hybrid model. We do not optimize for $\mathbf{F}_k$ any further. We obtain measurements $\mathbf{y}_k = \mathbf{H}\mathbf{x}_k + \mathbf{r}$ with $\mathbf{r} \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$. $\mathbf{H}$ is a sparse selection matrix that returns a two-dimensional position vector. We run this experiment in a filtering setting, i.e., we only use past observations. Therefore, we run the models as described in section 5 and section 6. For the Kalman filter baseline, we fit its transition covariance estimate $\hat{\mathbf{Q}} = s\mathbf{I}$ for s using ground truth data.

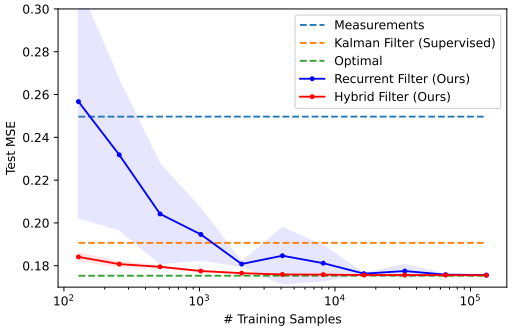

Figure 5: Results of the filtering models presented in this work. Given enough data, the self-supervised models approximate the optimal solution arbitrarily well. However, in the low-data regime our hybrid model significantly outperforms both the Kalman filter and the inference model.

In fig. 5 we depict the test error as a function of the number of training samples. The take-aways from the figure are two-fold. First it shows that in a self-supervised setting, we can get arbitrarily close to the optimal solution as long as we have enough data. This holds for both the pure inference model and the hybrid model. Second, using prior knowledge we out-perform the inference model in all data regimes, but significantly so in the low-data regime. Additionally, we report that the hybrid model's distance to the unobserved full states is much closer to the optimal solution than both that of the inference model and Kalman filter. Specifically, we report average mean squared errors of 0.685 for the inference model, 0.241 for the Kalman filter, which was fit using ground truth, $\mathbf{0.161}$ for the hybrid model compared to 0.135 for the optimal Kalman filter. Finally, it is worth noting that the hybrid model has much less variance as a function of its initialization.

7.2 Lotka-Volterra Dataset

The Lotka-Volterra equations describe the nonlinear dynamics of a biological system in which the population numbers of two species (predator-prey) interact. It is described by

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} = \begin{bmatrix} \alpha & -\beta x_1 \\ \gamma x_2 & -\delta \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{28}$$

where x_1 is the number of prey and x_2 the number of predators. We use $\mathbf{H} = \mathbf{I}$. Compared to the previous experiment, the system is now nonlinear. $\mathbf{F}_k$ again is obtained by a first-order Taylor expansion of the exponential of $\mathbf{A}_{|\mathbf{x}_k}$ evaluated at $\mathbf{x}_k$. We integrate this ODE at $dt = 0.001$ and select 4096 evenly-spaced time-steps at $\Delta t = 0.6$.

Model	MSE
Measurements	0.25
Krishnan et al., (2017)	0.18
Extended Kalman Smoother (Supervised)	0.12
Hybrid Filter + Linearized Smoother (Ours)	0.10
Hybrid Smoother (Ours)	0.07
Recurrent Smoother (Ours)	0.07

Table 1: Test error of our models on the Lotka-Volterra dataset (lower is better). Our inference and hybrid models outperform the baselines.

The results are shown in table 1. We compare against Krishnan et al. [2017] and the extended Kalman Filter [Ljung, 1979]. To allow for a fair comparison, we make sure that the generative model of Krishnan et al. [2017] makes use of the known $p(\mathbf{y}_k \mid \mathbf{x}_k)$, like our models do. In terms of test error,

our models perform better or comparably to these baselines. The Hybrid models do not outperform the recurrent model in this case. We hypothesize that this is because the first-order approximation that is used in the hybrid model can cripple the model when enough data is available to not rely on expert knowledge. Finally, we want to note that we believe the model by Krishnan et al. [2017] should be able to outperform the extended Kalman smoother. However, the variational approach has more (hyper-)parameters to tune and in our experiments we have not found the settings that lead to better performance.

7.3 Lorenz Equations

We simulate a Lorenz system according to

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} = \begin{bmatrix} -\sigma & \sigma & 0 \\ \rho - x_1 & -1 & 0 \\ x_2 & 0 & -\beta \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}. \quad (29)$$

We integrate the system using $dt = 0.00001$ and sample it evenly at $\Delta t = 0.05$. We have $\mathbf{H} = \mathbf{I}$ and thus $\mathbf{x} \in \mathbb{R}^3$ and $\mathbf{y} \in \mathbb{R}^3$. The Lorenz equations model atmospheric convection and form a classical example of chaos. Therefore, performing inference is much harder than in the linear case. We perform smoothing. $\mathbf{F}_k$ is linearly approximated by a Taylor expansion and used in the Kalman smoother and hybrid models. The baselines include the Extended Kalman smoother [Ljung, 1979], the hybrid model of Satorras et al. [2019] and the variational inference approach of Krishnan et al. [2017]. Our models include the recurrent model (section 5), the hybrid smoother (section 6), and the hybrid filter with linearized smoothing (section 6.2). For the supervised extended Kalman filter, we again optimize its covariance estimate $\hat{\mathbf{Q}}$ using ground truth data, and for the unsupervised one we maximize the data likelihood for $\hat{\mathbf{Q}}$.

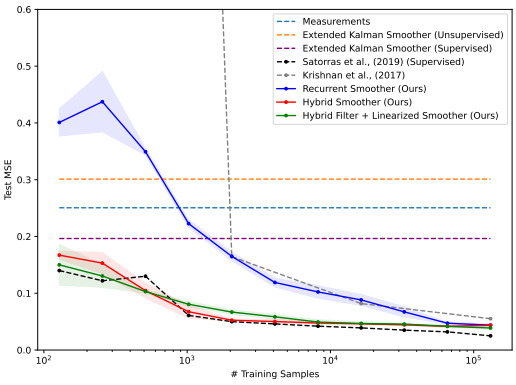

Figure 6: Lorenz system results. The self-supervised models out-perform the noisy measurements, the extended Kalman smoother and the variational approach.

We analyze the performance of the models under different data availability regimes. The results are shown in fig. 6 and qualitatively in fig. 1. It is clear that we (the recurrent model and hybrid model) approach the ground truth states with more data. This is in contrast to the Extended Kalman smoother, which barely outperforms the noisy measurements. We also see that the hybrid models significantly outperform the inference model in the low-data regime, making efficient use of the expert knowledge encoded in $\mathbf{F}_k$. Also notably, the Hybrid filter + linearized smoothing performs comparably to the full hybrid smoother, and better in low-data regimes. We hypothesize that this is the case since the required assumption for the linearized smoother holds (section 6.2) and regularizes the model. Again, the variational method Krishnan et al. [2017] performs poorly in low-data regimes. We believe the performance can be significantly improved with better tuning of this model. The supervised method by Satorras et al. [2019] outperforms our models only slightly.

8 Conclusion

We presented an advantageous recurrent parameterization of an inference model for nonlinear state-space models with higher-order latent Markov chains. This allows for efficient maximum-likelihood optimization of a self-supervised objective. Additionally, we introduce a recursive model in which we split the inference distribution into linear and nonlinear parts. The recursive approach introduces an optimization procedure that performs inference akin to the Kalman filter and smoother algorithms, but backpropgates through the updates. In this model, we can also incorporate domain knowledge (as it is also usually done in the Kalman filter) in a straightforward manner. For smoothing, we additionally propose a linearized Rauch-Tung-Striebel smoothing that can directly be applied to the (nonlinearly) parameterized filtering distributions. Experiments show that our models perform better or on par with fully supervised methods and outperform a method based on variational inference, especially in the low data regime.

References

Ienkaran Arasaratnam and Simon Haykin. Cubature kalman filters. *IEEE Transactions on automatic control*, 54(6):1254–1269, 2009.

Evan Archer, Il Memming Park, Lars Buesing, John Cunningham, and Liam Paninski. Black box variational inference for state space models. *arXiv preprint arXiv:1511.07367*, 2015.

Philipp Bader, Sergio Blanes, and Fernando Casas. Computing the matrix exponential with an optimized taylor polynomial approximation. *Mathematics*, 7(12):1174, 2019.

Joshua Batson and Loïc Royer. Noise2self: Blind denoising by self-supervision. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *ICML*, 2019.

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

Philipp Becker, Harit Pandya, Gregor H. W. Gebhardt, Cheng Zhao, C. James Taylor, and Gerhard Neumann. Recurrent kalman networks: Factorized inference in high-dimensional deep feature spaces. In *ICML*, 2019.

Katherine L Bouman, Michael D Johnson, Daniel Zoran, Vincent L Fish, Sheperd S Doeleman, and William T Freeman. Computational imaging for vlbi image reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 913–922, 2016.

Mark Briers, Arnaud Doucet, and Simon Maskell. Smoothing algorithms for state–space models. *Annals of the Institute of Statistical Mathematics*, 62(1):61, 2010.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C. Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *NeurIPS*, 2015.

SW Ellingson, GB Taylor, J Craig, J Hartman, J Dowell, CN Wolfe, TE Clarke, BC Hicks, NE Kassim, PS Ray, et al. The lwa1 radio telescope. *IEEE Transactions on Antennas and Propagation*, 61(5): 2540–2549, 2013.

Christian Gourieroux, Alain Monfort, and Alain Trognon. Pseudo maximum likelihood methods: Theory. *Econometrica: journal of the Econometric Society*, pages 681–700, 1984.

Tuomas Haarnoja, Anurag Ajay, Sergey Levine, and Pieter Abbeel. Backprop KF: learning discriminative deterministic state estimators. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *NeurIPS*, 2016.

Matthew J. Johnson, David Duvenaud, Alexander B. Wiltschko, Ryan P. Adams, and Sandeep R. Datta. Composing graphical models with neural networks for structured representations and fast inference. In *NeurIPS*, 2016.

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

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 *ICLR*, 2017.

Genshiro Kitagawa. The two-filter formula for smoothing and an implementation of the gaussian-sum smoother. *Annals of the Institute of Statistical Mathematics*, 46(4):605–623, 1994.

Genshiro Kitagawa. Monte carlo filter and smoother for non-gaussian nonlinear state space models. *Journal of computational and graphical statistics*, 5(1):1–25, 1996.

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

Rahul G. Krishnan, Uri Shalit, and David A. Sontag. Structured inference networks for nonlinear state space models. In *AAAI*, pages 2101–2109. AAAI Press, 2017.

Alexander Krull, Tim-Oliver Buchholz, and Florian Jug. Noise2void - learning denoising from single noisy images. In *CVPR*, 2019.

Mark Kuiack, Folkert Huizinga, Gijs Molenaar, Peeyush Prasad, Antonia Rowlinson, and Ralph AMJ Wijers. Aartfaac flux density calibration and northern hemisphere catalogue at 60 mhz. *Monthly Notices of the Royal Astronomical Society*, 482(2):2502–2514, 2019.

Samuli Laine, Tero Karras, Jaakko Lehtinen, and Timo Aila. High-quality self-supervised deep image denoising. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *NeurIPS*, 2019.

Wu Lin, Nicolas Hubacher, and Mohammad Emtiyaz Khan. Variational message passing with structured inference networks. In *ICLR (Poster)*. OpenReview.net, 2018.

Jun S Liu and Rong Chen. Sequential monte carlo methods for dynamic systems. *Journal of the American statistical association*, 93(443):1032–1044, 1998.

Lennart Ljung. Asymptotic behavior of the extended kalman filter as a parameter estimator for linear systems. *IEEE Transactions on Automatic Control*, 24(1):36–50, 1979.

Kevin Murphy and Stuart Russell. Rao-blackwellised particle filtering for dynamic bayesian networks. In *Sequential Monte Carlo methods in practice*, pages 499–515. Springer, 2001.

Syama Sundar Rangapuram, Matthias W. Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. Deep state space models for time series forecasting. In *NeurIPS*, pages 7796–7805, 2018.

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

David Ruhe, Mark Kuiack, Antonia Rowlinson, Ralph Wijers, and Patrick Forré. A real-time detection pipeline for transient sources in radio images using convolutional neural networks, 2021.

Victor Garcia Satorras, Max Welling, and Zeynep Akata. Combining generative and discriminative models for hybrid inference. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *NeurIPS*, 2019.

Bernhard Schölkopf, David W Hogg, Dun Wang, Daniel Foreman-Mackey, Dominik Janzing, Carl-Johann Simon-Gabriel, and Jonas Peters. Modeling confounding by half-sibling regression. *Proceedings of the National Academy of Sciences*, 113(27):7391–7398, 2016.

Richard A Thompson, James M Moran, and George W Swenson Jr. *Interferometry and synthesis in radio astronomy*. Springer Nature, 2017.

Eric A Wan and Rudolph Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *IEEE*, 2000.

A Gaussian Conditioning Formulas

Since many of the calculations used in this work are based on the Gaussian conditioning formulas, we provide them here. If

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}, \mathbf{P}) \tag{30}$$

$$p(\mathbf{y} \mid \mathbf{x}) = \mathcal{N}(\mathbf{y} \mid \mathbf{H}\mathbf{x} + \mathbf{b}, \mathbf{R}), \tag{31}$$

then

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y} \mid \mathbf{H}\boldsymbol{\mu} + \mathbf{b}, \mathbf{R} + \mathbf{H}\mathbf{P}\mathbf{H}^\top) \tag{32}$$

$$p(\mathbf{x} \mid \mathbf{y}) = \mathcal{N}\left(\mathbf{x} \mid \boldsymbol{\Sigma}\left[\mathbf{H}^\top \mathbf{R}^{-1}(\mathbf{y} - \mathbf{b}) + \mathbf{P}^{-1}\boldsymbol{\mu}\right], \boldsymbol{\Sigma}\right) \tag{33}$$

with

$$\boldsymbol{\Sigma} = (\mathbf{P}^{-1} + \mathbf{H}^\top \mathbf{R}^{-1}\mathbf{H})^{-1}. \tag{34}$$

B Higher Order Markov Graphical Model

Our model can be applied to state-space models with higher order Markov chains in latent space. An example was already given in fig. 2. In fig. 7 we exemplify a second-order Markov graphical model. This one has skip-connections in latent space.

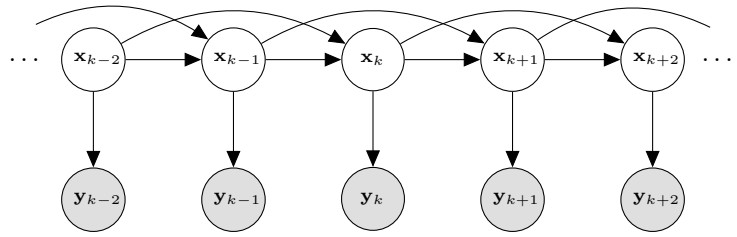

Figure 7: State-space model with second-order Markov chain in latent space.

C Bias-Variance-Noise Decomposition of the Self-Supervised Generalization Error

Any estimate $\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k(\mathbf{y}_{\neg k})$ for $\mathbf{x}_k$ that is not dependent on $\mathbf{y}_k$ will give us a bias-variance-noise decomposition of the generalization error. Note that this setting covers both the filtering and smoothing case. Define "optimal model" $\hat{\mathbf{y}}_k^* := \mathbb{E}\left[\mathbf{y}_k \mid \mathbf{y}_{\neg k}\right]$, then under the specified generative model (section 2) we have

$$\mathbb{E}\left[\|\hat{\mathbf{y}}_k - \mathbf{y}_k\|_2^2 \mid \mathbf{y}_{\neg k}\right] = \mathbb{E}\left[\|\hat{\mathbf{y}}_k - \hat{\mathbf{y}}_k^* + \hat{\mathbf{y}}_k^* - \mathbf{y}_k\|_2^2 \mid \mathbf{y}_{\neg k}\right] \tag{35}$$

$$= \mathbb{E}\left[\|\hat{\mathbf{y}}_k - \hat{\mathbf{y}}_k^*\|_2^2 \mid \mathbf{y}_{\neg k}\right] + \mathrm{Var}\left[\mathbf{y}_k \mid \mathbf{y}_{\neg k}\right] + 2\,\mathbb{E}\left[(\hat{\mathbf{y}}_k - \hat{\mathbf{y}}_k^*)^\top (\hat{\mathbf{y}}_k^* - \mathbf{y}_k) \mid \mathbf{y}_{\neg k}\right] \tag{36}$$

$$= \|\hat{\mathbf{y}}_k - \hat{\mathbf{y}}_k^*\|_2^2 + \mathrm{Var}\left[\mathbf{y}_k \mid \mathbf{y}_{\neg k}\right] + 2\,(\hat{\mathbf{y}}_k - \hat{\mathbf{y}}_k^*)^\top \underbrace{\mathbb{E}\left[(\hat{\mathbf{y}}_k^* - \mathbf{y}_k) \mid \mathbf{y}_{\neg k}\right]}_{0} \tag{37}$$

$$\mathrm{Var}\left[\mathbf{y}_k \mid \mathbf{y}_{\neg k}\right] = \mathbb{E}\left[\|\hat{\mathbf{y}}_k^* - \mathbf{H}\mathbf{x}_k - \mathbf{n}_k\|_2^2 \mid \mathbf{y}_{\neg k}\right] \tag{38}$$

$$= \mathbb{E}\left[\|\hat{\mathbf{y}}_k^* - \mathbf{H}\mathbf{x}_k\|_2^2 \mid \mathbf{y}_{\neg k}\right] + \mathbb{E}\left[\|\mathbf{n}_k\|_2^2 \mid \mathbf{y}_{\neg k}\right] + 2\underbrace{\mathbb{E}\left[(\mathbf{H}\mathbf{x}_k - \hat{\mathbf{y}}_k^*)^\top \mathbf{n}_k \mid \mathbf{y}_{\neg k}\right]}_{0} \tag{39}$$

$$\overset{\mathbf{x}_k, \mathbf{y}_{\neg k} \perp\!\!\!\perp \mathbf{n}_k}{=} \mathbb{E}\left[\|\hat{\mathbf{y}}_k^* - \mathbf{H}\mathbf{x}_k\|_2^2 \mid \mathbf{y}_{\neg k}\right] + \mathrm{tr}(\mathbf{R}) \tag{40}$$

Thus,

$$\mathbb{E}\left[\|\hat{\mathbf{y}}_k - \mathbf{y}_k\|_2^2 \mid \mathbf{y}_{\neg k}\right] = \|\hat{\mathbf{y}}_k - \hat{\mathbf{y}}_k^*\|_2^2 + \mathbb{E}\left[\|\hat{\mathbf{y}}_k^* - \mathbf{H}\mathbf{x}_k\|_2^2 \mid \mathbf{y}_{\neg k}\right] + \mathrm{tr}(\mathbf{R}) \tag{41}$$

Note that $\hat{\mathbf{y}}_k^* = \mathbb{E}\left[\mathbf{H}\mathbf{x}_k \mid \mathbf{y}_{\neg k}\right] = \mathbf{H}\,\mathbb{E}\left[\mathbf{x}_k \mid \mathbf{y}_{\neg k}\right]$. Then, define our model $\hat{\mathbf{y}}_k := \mathbf{H}\hat{\mathbf{x}}_k$. The *reducible* part of the error becomes

$$\|\hat{\mathbf{y}}_k^* - \hat{\mathbf{y}}_k\|_2^2 = \|\mathbf{H}\left(\mathbb{E}[\mathbf{x}_k \mid \mathbf{y}_{\neg k}] - \hat{\mathbf{x}}_k\right)\|_2^2 \tag{42}$$

For this reason, the model output $\hat{\mathbf{x}}_k$ approaches the optimal model under minimization of the self-supervised error (perturbed by $\mathbf{H}$).

Additionally, the model $\hat{\mathbf{y}}_k$ approaches $\mathbf{H}\mathbf{x}_k$ (the uncorrupted measurement) under this criterion.

$$\mathbb{E}\left[\left\|\hat{\mathbf{y}}_k - \mathbf{y}_k\right\|^2\right] = \mathbb{E}\left[\left\|\hat{\mathbf{y}}_k - \mathbf{H}\mathbf{x}_k + \mathbf{H}\mathbf{x}_k - \mathbf{y}_k\right\|^2\right] \tag{43}$$

$$= \mathbb{E}\left[\left\|\hat{\mathbf{y}}_k - \mathbf{H}\mathbf{x}_k\right\|^2\right] + \mathbb{E}\left[\left\|\mathbf{H}\mathbf{x}_k - \mathbf{y}_k\right\|^2\right] + 2\,\mathbb{E}\left[\left(\hat{\mathbf{y}}_k - \mathbf{H}\mathbf{x}_k\right)^\top \left(\mathbf{H}\mathbf{x}_k - \mathbf{y}_k\right)\right] \tag{44}$$

$$= \mathbb{E}\left[\left\|\hat{\mathbf{y}}_k - \mathbf{H}\mathbf{x}_k\right\|^2\right] + \mathbb{E}\left[\left\|\mathbf{H}\mathbf{x}_k - \mathbf{y}_k\right\|^2\right] + 2\,\mathbb{E}\left[\left(\hat{\mathbf{y}}_k - \mathbf{H}\mathbf{x}_k\right)\right]^\top \underbrace{\mathbb{E}\left[\mathbf{H}\mathbf{x}_k - \mathbf{y}_k\right]}_{0} \tag{45}$$

$$= \mathbb{E}\left[\left\|\hat{\mathbf{y}}_k - \mathbf{H}\mathbf{x}_k\right\|^2\right] + \operatorname{tr}(\mathbf{R}) + \underbrace{\mathbb{E}\left[\left\|\mathbf{H}\mathbf{x}_k - \mathbf{y}_k\right\|\right]^2}_{0} \tag{46}$$

If we then take $\hat{\mathbf{y}}_k := \mathbf{H}\hat{\mathbf{x}}_k$ then the reducible part of the error approximates the true $\mathbf{x}_k$ (perturbed by $\mathbf{H}$).

$$\mathbb{E}\left[\left\|\hat{\mathbf{y}}_k - \mathbf{H}\mathbf{x}_k\right\|^2\right] = \mathbb{E}\left[\left\|\mathbf{H}\left(\hat{\mathbf{x}}_k - \mathbf{x}_k\right)\right\|^2\right] \tag{47}$$

D Parameterized Smoothing

In section 3 we showed how we maximize the data likelihood in a filtering setting. Here, we show how that can be done for smoothing.

D.1 Parameterized Smoothing: Fitting

For the smoothing case we would, similar to before, parameterize:

$$p(\mathbf{x}_k \mid \mathbf{y}_{\neg k}) = \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{x}}_{k|\neg k}(\mathbf{y}_{\neg k}), \hat{\mathbf{P}}_{k|\neg k}(\mathbf{y}_{\neg k})\right), \tag{48}$$

where here we now use some form of two-sided recurrent neural network for $\hat{\mathbf{x}}_{k|\neg k}(\mathbf{y}_{\neg k})$ and $\hat{\mathbf{P}}_{k|\neg k}(\mathbf{y}_{\neg k})$, see section 5 and section 6. For fitting these models to the data we now would use a maximum-pseudo-likelihood [Gourieroux et al., 1984] approach by maximizing:

$$\sum_{k=0}^{K} \log p(\mathbf{y}_k \mid \mathbf{y}_{\neg k}) = \sum_{k=0}^{K} \log \mathcal{N}\left(\mathbf{y}_k \mid \mathbf{H}\,\hat{\mathbf{x}}_{k|\neg k}(\mathbf{y}_{\neg k}), \mathbf{H}\,\hat{\mathbf{P}}_{k|\neg k}(\mathbf{y}_{\neg k})\,\mathbf{H}^\top + \mathbf{R}\right), \tag{49}$$

leading to minimizing the following self-supervised loss function:

$$\mathcal{L} := \sum_{k=0}^{K} \left[\left(\hat{\mathbf{y}}_{k|\neg k} - \mathbf{y}_k\right)^\top \hat{\mathbf{M}}_{k|\neg k}^{-1} \left(\hat{\mathbf{y}}_{k|\neg k} - \mathbf{y}_k\right) + \log\det \hat{\mathbf{M}}_{k|\neg k}\right], \tag{50}$$

where $\hat{\mathbf{y}}_{k|\neg k} := \mathbf{H}\,\hat{\mathbf{x}}_{k|\neg k}(\mathbf{y}_{\neg k})$ and $\hat{\mathbf{M}}_{k|\neg k} := \mathbf{H}\,\hat{\mathbf{P}}_{k|\neg k}(\mathbf{y}_{\neg k})\,\mathbf{H}^\top + \mathbf{R}$.

D.2 Parameterized Smoothing: Improving the Inference

For improving the inference by also including $\mathbf{y}_k$ the formulas analogous to the ones in section 4 hold in the smoothing case as well. By just replacing $\mathbf{y}_{<k}$ before by $\mathbf{y}_{\neg k}$ and using eq. (48), $p(\mathbf{x}_k \mid \mathbf{y}_{\neg k}) = \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{x}}_{k|\neg k}(\mathbf{y}_{\neg k}), \hat{\mathbf{P}}_{k|\neg k}(\mathbf{y}_{\neg k})\right)$, we then get:

$$p(\mathbf{x}_k \mid \mathbf{y}_{0:K}) = \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{x}}_{k|0:K}(\mathbf{y}_{0:K}), \hat{\mathbf{P}}_{k|0:K}(\mathbf{y}_{\neg k})\right), \tag{51}$$

where we here use the following abbreviations:

$$\hat{\mathbf{P}}_{k|0:K}(\mathbf{y}_{\neg k}) := \left(\hat{\mathbf{P}}_{k|\neg k}^{-1} + \mathbf{H}^\top \mathbf{R}^{-1} \mathbf{H}\right)^{-1}, \tag{52}$$

$$\hat{\mathbf{x}}_{k|0:K}(\mathbf{y}_{0:K}) := \left(\hat{\mathbf{P}}_{k|\neg k}^{-1} + \mathbf{H}^\top \mathbf{R}^{-1} \mathbf{H}\right)^{-1} \left(\mathbf{H}^\top \mathbf{R}^{-1}\,\mathbf{y}_k + \hat{\mathbf{P}}_{k|\neg k}^{-1}\,\hat{\mathbf{x}}_{k|\neg k}\right). \tag{53}$$

We can similarly use the Kalman gain matrix from eq. (13) and eq. (12) if $M \leq N$:

$$\hat{\mathbf{K}}_{k|\neg k} := \hat{\mathbf{P}}_{k|\neg k}\,\mathbf{H}^\top \left(\mathbf{H}\,\hat{\mathbf{P}}_{k|\neg k}\,\mathbf{H}^\top + \mathbf{R}\right)^{-1}, \tag{54}$$

$$\hat{\mathbf{x}}_{k|0:K} = \hat{\mathbf{x}}_{k|\neg k} + \hat{\mathbf{K}}_{k|\neg k}\left(\mathbf{y}_k - \mathbf{H}\,\hat{\mathbf{x}}_{k|\neg k}\right), \tag{55}$$

$$\hat{\mathbf{P}}_{k|0:K} = \hat{\mathbf{P}}_{k|\neg k} - \hat{\mathbf{K}}_{k|\neg k}\,\mathbf{H}\,\hat{\mathbf{P}}_{k|\neg k}. \tag{56}$$

E Parameterized Hybrid Smoothing

In analogy to the hybrid filter model section 6.1 we can use a hybrid parameterization for the smoothing case:

$$p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{\neg k}) = \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{F}}_{k|\neg k}\, \mathbf{x}_{k-1} + \hat{\mathbf{e}}_{k|\neg k}, \hat{\mathbf{Q}}_{k|\neg k}\right), \tag{57}$$

where $\hat{\mathbf{F}}_{k|\neg k}(\mathbf{y}_{\neg k})$ and $\hat{\mathbf{e}}_{k|\neg k}(\mathbf{y}_{\neg k})$ and $\hat{\mathbf{Q}}_{k|\neg k}(\mathbf{y}_{\neg k})$ are two-sided recurrent neural network outputs similar to section 5. We compute the distribution of interest as follows:

$$\begin{aligned}
p(\mathbf{x}_k \mid \mathbf{y}_{0:K}) &= \frac{p(\mathbf{y}_k \mid \mathbf{x}_k) p(\mathbf{x}_k \mid \mathbf{y}_{\neg k})}{p(\mathbf{y}_k \mid \mathbf{y}_{\neg k})} && (58)\\
&\propto p(\mathbf{y}_k \mid \mathbf{x}_k) \int p(\mathbf{x}_k, \mathbf{x}_{k-1} \mid \mathbf{y}_{\neg k}) d\mathbf{x}_{k-1} && (59)\\
&= p(\mathbf{y}_k \mid \mathbf{x}_k) \int p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{\neg k}) p(\mathbf{x}_{k-1} \mid \mathbf{y}_{\neg k}) d\mathbf{x}_{k-1} && (60)\\
&\overset{\mathbf{x}_{k-1} \perp\!\!\!\perp \mathbf{y}_k \mid \mathbf{y}_{\neg k}}{\approx} p(\mathbf{y}_k \mid \mathbf{x}_k) \int p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{\neg k}) p(\mathbf{x}_{k-1} \mid \mathbf{y}_{0:K}) d\mathbf{x}_{k-1} && (61)
\end{aligned}$$

where we make the approximation to compute eq. (58) efficiently and recursively. It is justified if $I(\mathbf{x}_{k-1}; \mathbf{y}_k \mid \mathbf{y}_{\neg k}) < \epsilon$ for small ϵ. That is, the additional information that $\mathbf{y}_k$ conveys about $\mathbf{x}_{k-1}$ is marginal if we have all other data. Let

$$p(\mathbf{x}_{k-1} \mid \mathbf{y}_{0:K}) := \mathcal{N}\left(\mathbf{x}_{k-1} \mid \hat{\mathbf{x}}_{k-1|0:K}, \hat{\mathbf{P}}_{k-1|0:K}\right) \tag{62}$$

be previous time-step's posterior. Then put

$$p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{\neg k}) := \mathcal{N}(\mathbf{x} \mid \hat{\mathbf{F}}_{k|\neg k}\mathbf{x}_{k-1} + \hat{\mathbf{e}}_{k|\neg k}, \hat{\mathbf{Q}}_{k|\neg k}). \tag{63}$$

where we left out the arguments for the following quantities estimated by an RNN.

$$\begin{bmatrix} \mathbf{g}_k^{\rightarrow} \\ \mathbf{g}_k^{\leftarrow} \end{bmatrix} := \begin{bmatrix} \mathbf{y}_{k-1} - \mathbf{y}_{k-2} \\ \mathbf{y}_{k+2} - \mathbf{y}_{k+1} \end{bmatrix} \quad \begin{bmatrix} \mathbf{h}_k^{\rightarrow} \\ \mathbf{h}_k^{\leftarrow} \end{bmatrix} := \begin{bmatrix} \mathrm{GRU}(\mathbf{h}_{k-1}^{\rightarrow}, \mathbf{g}_k^{\rightarrow}) \\ \mathrm{GRU}(\mathbf{h}_{k+1}^{\leftarrow}, \mathbf{g}_k^{\leftarrow}) \end{bmatrix} \begin{bmatrix} \hat{\mathbf{F}}_{k|\neg k}(\mathbf{y}_{\neg k}) \\ \hat{\mathbf{e}}_{k|\neg k}(\mathbf{y}_{\neg k}) \\ \hat{\mathbf{Q}}_{k|\neg k}(\mathbf{y}_{\neg k}) \end{bmatrix} := \boldsymbol{f}(\mathbf{h}_k^{\rightarrow}, \mathbf{h}_k^{\leftarrow}) \tag{64}$$

Applying the integral in eq. (61) we get

$$\int p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{\neg k}) p(\mathbf{x}_{k-1} \mid \mathbf{y}_{0:K}) d\mathbf{x}_{k-1} = \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{x}}_{k|\neg k}, \hat{\mathbf{P}}_{k|\neg k}\right) \tag{65}$$

where we put

$$\hat{\mathbf{x}}_{k|\neg k} := \hat{\mathbf{F}}_{k|\neg k}\hat{\mathbf{x}}_{k-1|0:K} + \hat{\mathbf{e}}_{k|\neg k} \qquad \hat{\mathbf{P}}_{k|\neg k} := \hat{\mathbf{F}}_{k|\neg k}\hat{\mathbf{P}}_{k-1|0:K}\hat{\mathbf{F}}_{k|\neg k}^{\top} + \hat{\mathbf{Q}}_{k|\neg k} \tag{66}$$

The distribution of interest eq. (58) can then be computed using the Gaussian conditioning formulas in the same way as the parameterized recurrent smoother eqs. (54) to (56).

The likelihood term can be computed as follows

$$\begin{aligned}
p(\mathbf{y}_k \mid \mathbf{y}_{\neg k}) &= \int p(\mathbf{y}_k \mid \mathbf{x}_k) p(\mathbf{x}_k \mid \mathbf{y}_{\neg k}) d\mathbf{x}_k && (67)\\
&= \int p(\mathbf{y}_k \mid \mathbf{x}_k) \int p(\mathbf{x}_{k-1}, \mathbf{x}_k \mid \mathbf{y}_{\neg k}) d\mathbf{x}_k \mathbf{x}_{k-1} && (68)\\
&= \int p(\mathbf{y}_k \mid \mathbf{x}_k) \int p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{\neg k}) p(\mathbf{x}_{k-1} \mid \mathbf{y}_{\neg k}) d\mathbf{x}_k \mathbf{x}_{k-1} && (69)\\
&\overset{\mathbf{x}_{k-1} \perp\!\!\!\perp \mathbf{y}_k \mid \mathbf{y}_{\neg k}}{\approx} \int p(\mathbf{y}_k \mid \mathbf{x}_k) \int p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{\neg k}) p(\mathbf{x}_{k-1} \mid \mathbf{y}_{0:K}) d\mathbf{x}_k \mathbf{x}_{k-1} && (70)
\end{aligned}$$

where we made the same approximation eq. (61) as before. It evaluates to

$$\int p(\mathbf{y}_k \mid \mathbf{x}_k) p(\mathbf{x}_k \mid \mathbf{y}_{\neg k}) d\mathbf{x}_k = \mathcal{N}\left(\mathbf{y}_k \mid \hat{\mathbf{y}}_{k|\neg k}, \hat{\mathbf{M}}_{k|\neg k}\right) \tag{71}$$

with

$$\hat{\mathbf{y}}_{k|\neg k} := \mathbf{H}\hat{\mathbf{x}}_{k|\neg k} \qquad \hat{\mathbf{M}}_{k|\neg k} := \mathbf{H}\hat{\mathbf{P}}_{k|\neg k}\mathbf{H}^{\top} + \mathbf{R} \tag{72}$$

E.1 Parameterized Hybrid Smoother: Alternative Posterior Evaluation

We would like to point out that the distribution $p(\mathbf{x}_k \mid \mathbf{y}_{0:K})$ can be obtained without making assumption eq. (61), which we stretch out here. Initial experiments showed that using these calculations the model did not converge as smoothly as when using the ones stated before. However, it could be of interest to further investigate. Returning to the posterior of interest

$$p(\mathbf{x}_k \mid \mathbf{y}_{0:K}) = \int p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{0:K})\, p(\mathbf{x}_{k-1} \mid \mathbf{y}_{0:K})\, d\mathbf{x}_{k-1} \tag{73}$$

$$= \int \frac{p(\mathbf{y}_k \mid \mathbf{x}_k)}{p(\mathbf{y}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{\neg k})}\, p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{\neg k}) p(\mathbf{x}_{k-1} \mid \mathbf{y}_{0:K})\, d\mathbf{x}_{k-1}. \tag{74}$$

We have

$$p(\mathbf{x}_{k-1} \mid \mathbf{y}_{0:K}) = \mathcal{N}\left(\mathbf{x}_{k-1} \mid \hat{\mathbf{x}}_{k-1|0:K}(\mathbf{y}_{0:K}), \hat{\mathbf{P}}_{k-1|0:K}(\mathbf{y}_{0:K})\right) \tag{75}$$

as the previous time-step's posterior.

$$p\left(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{\neg k}\right) = \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{F}}_{k|\neg k}\mathbf{x}_{k-1} + \hat{\mathbf{e}}_{k|\neg k}, \hat{\mathbf{Q}}_{k|\neg k}\right) \tag{76}$$

where $\hat{\mathbf{F}}_{k|\neg k}(\mathbf{y}_{\neg k})$, $\hat{\mathbf{e}}_{k|\neg k}(\mathbf{y}_{\neg k})$ and $\hat{\mathbf{Q}}_{k|\neg k}(\mathbf{y}_{\neg k})$ are estimated by a neural network. Combining this with noise model eq. (2) we get:

$$p(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_{0:K})$$
$$= \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{F}}_{k|\neg k}\, \mathbf{x}_{k-1} + \hat{\mathbf{e}}_{k|\neg k} + \mathbf{K}_k\, (\mathbf{y}_k - \mathbf{H}\, (\hat{\mathbf{F}}_{k|\neg k}\, \mathbf{x}_{k-1} + \hat{\mathbf{e}}_{k|\neg k})), \hat{\mathbf{Q}}_{k|\neg k} - \mathbf{K}_k\, \mathbf{H}\, \hat{\mathbf{Q}}_{k|\neg k}\right) \tag{77}$$

$$= \mathcal{N}\left(\mathbf{x}_k \mid (\mathbf{I} - \mathbf{K}_n\, \mathbf{H})\, \left(\hat{\mathbf{F}}_{k|\neg k}\, \mathbf{x}_{k-1} + \hat{\mathbf{e}}_{k|\neg k}\right) + \mathbf{K}_k\, \mathbf{y}_k, (\mathbf{I} - \mathbf{K}_n\, \mathbf{H})\, \hat{\mathbf{Q}}_{k|\neg k}\right), \tag{78}$$

where we directly applied the Woodbury matrix identity to obtain Kalman gain matrix

$$\mathbf{K}_k := \hat{\mathbf{Q}}_{k|\neg k}\, \mathbf{H}^\top\, \left(\mathbf{H}\, \hat{\mathbf{Q}}_{k|\neg k}\, \mathbf{H}^\top + \mathbf{R}\right)^{-1} \tag{79}$$

Then, applying the integral we get:

$$p\left(\mathbf{x}_k \mid \mathbf{y}_{0:K}\right) = \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{x}}_{k|0:K}, \hat{\mathbf{P}}_{k|0:K}\right), \tag{80}$$

with:

$$\hat{\mathbf{P}}_{k|0:K} = (\mathbf{I} - \mathbf{K}_n\, \mathbf{H})\, \hat{\mathbf{F}}_{k|\neg k}\, \hat{\mathbf{P}}_{k-1|0:K}\, \hat{\mathbf{F}}_{k|\neg k}^\top\, (\mathbf{I} - \mathbf{K}_n\, \mathbf{H})^\top + (\mathbf{I} - \mathbf{K}_n\, \mathbf{H})\, \hat{\mathbf{Q}}_{k|\neg k}, \tag{81}$$

and

$$\hat{\mathbf{x}}_{k|0:K} = \hat{\mathbf{F}}_{k|\neg k}\hat{\mathbf{x}}_{k-1|0:K} + \hat{\mathbf{e}}_{k|\neg k} + \mathbf{K}_k\, (\mathbf{y}_k - \mathbf{H}\, (\hat{\mathbf{F}}_{k|\neg k}\hat{\mathbf{x}}_{k-1|0:K} + \hat{\mathbf{e}}_{k|\neg k})) \tag{82}$$

$$= (\mathbf{I} - \mathbf{K}_n\, \mathbf{H})\, \left(\hat{\mathbf{F}}_{k|\neg k}\, \hat{\mathbf{x}}_{k-1|0:K} + \hat{\mathbf{e}}_{k|\neg k}\right) + \mathbf{K}_k\, \mathbf{y}_k. \tag{83}$$

F Algorithms

Algorithm 1: Recurrent Smoother (Training)

input : Training data (time-series) $\mathbf{y}_{0:K} = (\mathbf{y}_0, \ldots, \mathbf{y}_K)$, emission function $\mathcal{N}(\mathbf{H}\mathbf{x}_k, \mathbf{R})$, initialized parameters ϕ_0, number of training iterations n.

output : Optimized parameters ϕ^*

for *i in 1 to n* **do**

$\mathbf{h}_0 := \mathbf{0}$

$\mathbf{h}_{K+1} := \mathbf{0}$

$\mathcal{L}^{(i)} := 0$

for *k in 0 to K* **do**

$$\mathbf{g}_k^{\rightarrow} := \mathbf{y}_{k-1} - \mathbf{y}_{k-2} \text{ if } k > 1 \text{ else } \mathbf{g}_k^{\rightarrow} := \mathbf{0}$$
$$\mathbf{g}_k^{\leftarrow} := \mathbf{y}_{k+1} - \mathbf{y}_{k+2} \text{ if } k < K-1 \text{ else } \mathbf{g}_k^{\leftarrow} := \mathbf{0}$$
$$\mathbf{h}_k^{\rightarrow} := \mathrm{GRU}_\phi(\mathbf{h}_{k-1}^{\rightarrow}, \mathbf{g}_k^{\rightarrow})$$
$$\mathbf{h}_k^{\leftarrow} := \mathrm{GRU}_\phi(\mathbf{h}_{k-1}^{\leftarrow}, \mathbf{g}_k^{\leftarrow})$$
$$\begin{bmatrix} \hat{\mathbf{x}}_{k|\neg k} \\ \hat{\mathbf{L}}_{k|\neg k} \end{bmatrix} := \boldsymbol{f}_\phi(\mathbf{h}_k^{\leftarrow}, \mathbf{h}_k^{\rightarrow})$$
$$\hat{\mathbf{P}}_{k|\neg k} := \hat{\mathbf{L}}_{k|\neg k}\hat{\mathbf{L}}_{k|\neg k}^\top$$
$$\hat{\mathbf{y}}_{k|\neg k} := \mathbf{H}\hat{\mathbf{x}}_{k|\neg k}$$
$$\hat{\mathbf{B}}_{k|\neg k} := \left(\mathbf{H}\hat{\mathbf{P}}_{k|\neg k}\mathbf{H}^\top + \mathbf{R}\right)^{-1}$$
$$\mathcal{L}_k^{(i)} := \mathcal{L}_{k-1}^{(i)} + (\mathbf{y}_k - \hat{\mathbf{y}}_{k|\neg k})^\top \hat{\mathbf{B}}_{k|\neg k} (\mathbf{y}_k - \hat{\mathbf{y}}_{k|\neg k}) - \log\det \hat{\mathbf{B}}_{k|\neg k}$$

end

Compute $\nabla_\phi \mathcal{L}_K^{(i)}$ and apply SGD step with respect to all model parameters, which amounts to backpropagation through the above calculations.

end

For the filter variant, the steps that involve the backward direction ($\leftarrow$) are left out.

Algorithm 2: Recurrent Smoother (Inference)

input : Test data $\mathbf{y}_{0:K} = (\mathbf{y}_0, \ldots, \mathbf{y}_K)$, trained parameters ϕ^*, emission function $\mathcal{N}(\mathbf{H}\mathbf{x}, \mathbf{R})$.

output : Inferred posteriors $p(\mathbf{x}_k \mid \mathbf{y}_{0:K}) = \mathcal{N}(\mathbf{x}_k \mid \hat{\mathbf{x}}_{k|0:K}, \hat{\mathbf{P}}_{k|0:K})$ for all k

$\mathbf{h}_0 := \mathbf{0}$

$\mathbf{h}_{K+1} := \mathbf{0}$

for *k in 0 to K* **do**

$$\mathbf{g}_k^{\rightarrow} := \mathbf{y}_{k-1} - \mathbf{y}_{k-2} \text{ if } k > 1 \text{ else } \mathbf{g}_k^{\rightarrow} := \mathbf{0}$$
$$\mathbf{g}_k^{\leftarrow} := \mathbf{y}_{k+1} - \mathbf{y}_{k+2} \text{ if } k < K-1 \text{ else } \mathbf{g}_k^{\leftarrow} := \mathbf{0}$$
$$\mathbf{h}_k^{\rightarrow} := \mathrm{GRU}_\phi(\mathbf{h}_{k-1}^{\rightarrow}, \mathbf{g}_k^{\rightarrow})$$
$$\mathbf{h}_k^{\leftarrow} := \mathrm{GRU}_\phi(\mathbf{h}_{k-1}^{\leftarrow}, \mathbf{g}_k^{\leftarrow})$$
$$\begin{bmatrix} \hat{\mathbf{x}}_{k|\neg k} \\ \hat{\mathbf{L}}_{k|\neg k} \end{bmatrix} := \boldsymbol{f}_\phi(\mathbf{h}_k^{\leftarrow}, \mathbf{h}_k^{\rightarrow})$$
$$\hat{\mathbf{P}}_{k|\neg k} := \hat{\mathbf{L}}_{k|\neg k}\hat{\mathbf{L}}_{k|\neg k}^\top$$
$$\hat{\mathbf{K}}_k := \hat{\mathbf{P}}_{k|\neg k}\mathbf{H}^\top \left(\mathbf{H}\hat{\mathbf{P}}_{k|\neg k}\mathbf{H}^\top + \mathbf{R}\right)^{-1}$$
$$\hat{\mathbf{x}}_{k|0:K} := \hat{\mathbf{x}}_{k|\neg k} + \hat{\mathbf{K}}_k(\mathbf{y}_k - \mathbf{H}\hat{\mathbf{x}}_{k|\neg k})$$
$$\hat{\mathbf{P}}_{k|0:K} := \hat{\mathbf{P}}_{k|\neg k} - \hat{\mathbf{K}}_k\mathbf{H}\hat{\mathbf{P}}_{k|\neg k}$$

end

For the filter variant, the steps that involve the backward direction ($\leftarrow$) are left out.

Algorithm 3: Hybrid Filter (Inference)

input : Data (time-series) $\mathbf{y}_{0:K} = (\mathbf{y}_0, \ldots, \mathbf{y}_K)$, emission matrices $\mathbf{H}$ and $\mathbf{R}$, parameters ϕ.
output :

1. For training: Loss value $\mathcal{L}_K$ and its gradient $\nabla_\phi \mathcal{L}_K$.
2. For inference: $\hat{\mathbf{x}}_{k|\leq k}$ and $\hat{\mathbf{P}}_{k|\leq k}$ for k in $0, \ldots, K$. Inference is done via: $p(\mathbf{x}_k \mid \mathbf{y}_{\leq k}) = \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{x}}_{k|\leq k}, \hat{\mathbf{P}}_{k|\leq k}\right)$.
3. For linearized smoothing (algorithm 5): $\hat{\mathbf{F}}_{k|<k}$, $\hat{\mathbf{P}}_{k|<k}$, $\hat{\mathbf{x}}_{k|\leq k}$ and $\hat{\mathbf{P}}_{k|\leq k}$, for k in $0, \ldots, K$.
4. For forecasting: $\hat{\mathbf{x}}_{K+1|<(K+1)}$ and $\hat{\mathbf{P}}_{K+1|<(K+1)}$. Forecasting is done via: $p(\mathbf{x}_{K+1} \mid \mathbf{y}_{0:K}) = \mathcal{N}\left(\mathbf{x}_{K+1} \mid \hat{\mathbf{x}}_{K+1|<(K+1)}, \hat{\mathbf{P}}_{K+1|<(K+1)}\right)$.

$\mathbf{h}_0 := \mathbf{0}$
$\mathcal{L}_{-1} := 0$
$\hat{\mathbf{P}}_{0|<0} := \hat{\mathbf{Q}}_0$
$\hat{\mathbf{x}}_{0|<0} := \hat{\mathbf{e}}_0$
for k *in* $0, \ldots, K$ **do**

$$\hat{\mathbf{B}}_{k|<k} := \left(\mathbf{H}\,\hat{\mathbf{P}}_{k|<k}\,\mathbf{H}^\top + \mathbf{R}\right)^{-1}$$
$$\hat{\mathbf{y}}_{k|<k} := \mathbf{H}\,\hat{\mathbf{x}}_{k|<k}$$
$$\mathcal{L}_k := \mathcal{L}_{k-1} + (\mathbf{y}_k - \hat{\mathbf{y}}_{k|<k})^\top\,\hat{\mathbf{B}}_{k|<k}\,(\mathbf{y}_k - \hat{\mathbf{y}}_{k|<k}) - \log\det\hat{\mathbf{B}}_{k|<k}$$
$$\hat{\mathbf{K}}_k := \hat{\mathbf{P}}_{k|<k}\,\mathbf{H}^\top\hat{\mathbf{B}}_{k|<k}$$
$$\hat{\mathbf{P}}_{k|\leq k} := \hat{\mathbf{P}}_{k|<k} - \hat{\mathbf{K}}_k\,\mathbf{H}\,\hat{\mathbf{P}}_{k|<k}$$
$$\hat{\mathbf{x}}_{k|\leq k} := \hat{\mathbf{x}}_{k|<k} + \hat{\mathbf{K}}_k\,(\mathbf{y}_k - \hat{\mathbf{y}}_{k|<k})$$
$$\mathbf{g}_{k+1} := \mathbf{y}_k - \mathbf{y}_{k-1} \text{ if } k > 1 \text{ else } \mathbf{g}_{k+1} := \mathbf{0}$$
$$\mathbf{h}_{k+1} := \mathrm{GRU}_\phi(\mathbf{h}_k, \mathbf{g}_{k+1})$$
$$\begin{bmatrix}\hat{\mathbf{e}}_{k+1|<(k+1)}\\ \hat{\mathbf{F}}_{k+1|<(k+1)}\\ \hat{\mathbf{L}}_{k+1|<(k+1)}\end{bmatrix} := \boldsymbol{f}_\phi(\mathbf{h}_{k+1})$$
$$\hat{\mathbf{Q}}_{k+1|<(k+1)} := \hat{\mathbf{L}}_{k+1|<(k+1)}\,\hat{\mathbf{L}}^\top_{k+1|<(k+1)}$$
$$\hat{\mathbf{P}}_{k+1|<(k+1)} := \hat{\mathbf{F}}_{k+1|<(k+1)}\,\hat{\mathbf{P}}_{k|\leq k}\,\hat{\mathbf{F}}^\top_{k+1|<(k+1)} + \hat{\mathbf{Q}}_{k+1|<(k+1)}$$
$$\hat{\mathbf{x}}_{k+1|<(k+1)} := \hat{\mathbf{F}}_{k+1|<(k+1)}\,\hat{\mathbf{x}}_{k|\leq k} + \hat{\mathbf{e}}_{k+1|<(k+1)}$$

end
For the training case we use backpropagation through the above loop to compute $\nabla_\phi \mathcal{L}_K$.

Algorithm 4: Hybrid Filter (Training)

input : Data (time-series) $\mathbf{y}_{0:K} = (\mathbf{y}_0, \ldots, \mathbf{y}_K)$, , emission matrices $\mathbf{H}$ and $\mathbf{R}$, initialized parameters ϕ_0, number of training rounds I.

output : Model parameters ϕ^* for inference at test-time.

for i *in* $0, \ldots, I$ **do**

  Obtain $\mathcal{L}_K^{(i)}$ and $\nabla_\phi \mathcal{L}_K^{(i)}$ from algorithm 3.

  Run preferred optimizer step to update parameters ϕ with $\nabla_\phi \mathcal{L}_K^{(i)}$ (and $\mathcal{L}_K^{(i)}$).

end

Algorithm 5: Linearized Smoother

input : Values of: $\hat{\mathbf{F}}_{k|<k}$, $\hat{\mathbf{P}}_{k|<k}$, $\hat{\mathbf{P}}_{k|\leq k}$, $\hat{\mathbf{x}}_{k|\leq k}$ for all $k = 0, \ldots, K$, obtained from the hybrid filter algorithm.

output : Linearly smoothed distributions $p(\mathbf{x}_k \mid \mathbf{y}_{0:K}) = \mathcal{N}\left(\mathbf{x}_k \mid \hat{\mathbf{z}}_k, \hat{\mathbf{G}}_k\right)$ for all $k = 0, \ldots, K$.

$\hat{\mathbf{z}}_K := \hat{\mathbf{x}}_{K|\leq K}$

$\hat{\mathbf{G}}_K := \hat{\mathbf{P}}_{K|\leq K}$

for $k = K, \ldots, 1$ **do**

$$\hat{\mathbf{J}}_{k-1|k} := \hat{\mathbf{P}}_{k-1|\leq k-1}\, \hat{\mathbf{F}}_{k|<k}^\top\, \hat{\mathbf{P}}_{k|<k}^{-1}$$

$$\hat{\mathbf{G}}_{k-1} := \hat{\mathbf{P}}_{k-1|\leq k-1} + \hat{\mathbf{J}}_{k-1|k} \left(\hat{\mathbf{P}}_{k|\leq k} - \hat{\mathbf{P}}_{k|<k}\right) \hat{\mathbf{J}}_{k-1|k}^\top$$

$$\hat{\mathbf{z}}_{k-1} := \hat{\mathbf{x}}_{k-1|\leq k-1} + \hat{\mathbf{J}}_{k-1|k} \left(\hat{\mathbf{z}}_k - \hat{\mathbf{x}}_{k|\leq k}\right)$$

end

Algorithm 6: Parameterized Hybrid Smoother (Inference)

input : Data (time-series) $\mathbf{y}_{0:K} = (\mathbf{y}_0, \ldots, \mathbf{y}_K)$, emission matrices $\mathbf{H}$ and $\mathbf{R}$, initialized parameters ϕ_0

output :

1. Loss value $\mathcal{L}_K$ and its gradient w.r.t. all model parameters $\nabla_\phi(\mathcal{L}_K)$ for training.
2. For all k in $0, \ldots, K$: $\hat{\mathbf{x}}_{k|0:K}, \hat{\mathbf{P}}_{k|0:K}$. These can be used for inference through $p(\mathbf{x}_k \mid \mathbf{y}_{0:K}) = \mathcal{N}\left(\hat{\mathbf{x}}_{k|0:K}, \hat{\mathbf{P}}_{k|0:K}\right)$.

$\overrightarrow{\mathbf{h}_0} := \mathbf{0}$
$\overleftarrow{\mathbf{h}_{K+1}} := \mathbf{0}$
$\mathcal{L}_{-1}^{(i)} := 0$
$\hat{\mathbf{P}}_{0|\neg 0} := \hat{\mathbf{Q}}_0$
$\hat{\mathbf{x}}_{0|\neg 0} := \hat{\mathbf{e}}_0$
for k *in* $0, \ldots, K$ **do**

$$\hat{\mathbf{B}}_{k|\neg k} := \left(\mathbf{H}\,\hat{\mathbf{P}}_{k|\neg k}\,\mathbf{H}^\top + \mathbf{R}\right)^{-1}$$
$$\hat{\mathbf{y}}_{k|\neg k} := \mathbf{H}\,\hat{\mathbf{x}}_{k|\neg k}$$
$$\mathcal{L}_k^{(i)} := \mathcal{L}_{k-1}^{(i)} + (\mathbf{y}_k - \hat{\mathbf{y}}_{k|\neg k})^\top\,\hat{\mathbf{B}}_{k|\neg k}\,(\mathbf{y}_k - \hat{\mathbf{y}}_{k|\neg k}) - \log\det\hat{\mathbf{B}}_{k|\neg k}$$
$$\hat{\mathbf{K}}_k := \hat{\mathbf{P}}_{k|\neg k}\,\mathbf{H}^\top\hat{\mathbf{B}}_{k|\neg k}$$
$$\hat{\mathbf{P}}_{k|\neg k} := \hat{\mathbf{P}}_{k|\neg k} - \hat{\mathbf{K}}_k\,\mathbf{H}\,\hat{\mathbf{P}}_{k|\neg k}$$
$$\hat{\mathbf{x}}_{k|\neg k} := \hat{\mathbf{x}}_{k|\neg k} + \hat{\mathbf{K}}_k\,(\mathbf{y}_k - \hat{\mathbf{y}}_{k|\neg k})$$
$$\overrightarrow{\mathbf{g}_{k+1}} := \mathbf{y}_k - \mathbf{y}_{k-1} \text{ if } k > 1 \text{ else } \overrightarrow{\mathbf{g}_{k+1}} := \mathbf{0}$$
$$\overleftarrow{\mathbf{g}_{k+1}} := \mathbf{y}_{k+2} - \mathbf{y}_{k+3} \text{ if } k+1 < K-2 \text{ else } \overleftarrow{\mathbf{g}_{k+1}} := \mathbf{0}$$
$$\overrightarrow{\mathbf{h}_{k+1}} := \mathrm{GRU}_\phi(\overrightarrow{\mathbf{h}_k}, \overrightarrow{\mathbf{g}_{k+1}})$$
$$\overleftarrow{\mathbf{h}_{k+1}} := \mathrm{GRU}_\phi(\overleftarrow{\mathbf{h}_k}, \overleftarrow{\mathbf{g}_{k+1}})$$
$$\begin{bmatrix}\hat{\mathbf{e}}_{k+1|\neg(k+1)} \\ \hat{\mathbf{F}}_{k+1|\neg(k+1)} \\ \hat{\mathbf{L}}_{k+1|\neg(k+1)}\end{bmatrix} := \boldsymbol{f}_\phi(\overrightarrow{\mathbf{h}_{k+1}}, \overleftarrow{\mathbf{h}_{k+1}})$$
$$\hat{\mathbf{Q}}_{k+1|\neg(k+1)} := \hat{\mathbf{L}}_{k+1|\neg(k+1)}\,\hat{\mathbf{L}}_{k+1|\neg(k+1)}^\top$$
$$\hat{\mathbf{P}}_{k+1|\neg(k+1)} := \hat{\mathbf{F}}_{k+1|\neg(k+1)}\,\hat{\mathbf{P}}_{k|0:K}\,\hat{\mathbf{F}}_{k+1|\neg(k+1)}^\top + \hat{\mathbf{Q}}_{k+1|\neg(k+1)}$$
$$\hat{\mathbf{x}}_{k+1|\neg(k+1)} := \hat{\mathbf{F}}_{k+1|\neg(k+1)}\,\hat{\mathbf{x}}_{k|0:K} + \hat{\mathbf{e}}_{k+1|\neg(k+1)}$$

end
For training case we also use backpropagation through the above loop to compute $\nabla_\phi\mathcal{L}_K$

Algorithm 7: Parameterized Hybrid Smoother (Training)

input : Data (time-series) $\mathbf{y}_{0:K} = (\mathbf{y}_0, \ldots, \mathbf{y}_K)$, , emission matrices $\mathbf{H}$ and $\mathbf{R}$, initialized parameters ϕ_0, number of training rounds I

output : Model parameters ϕ^* for inference at test-time.

for i *in* $1, \ldots, I$ **do**
  Obtain $\nabla_\phi\mathcal{L}_K^{(i)}$ from algorithm 6.
  Run preferred optimizer step.
end

G Experiments

G.1 Linear Dynamics

As specified in the main paper, the dynamics are according to

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -c & 1 \\ 0 & -\tau c & 0 \end{bmatrix} \begin{bmatrix} p \\ v \\ a \end{bmatrix}. \tag{84}$$

Since these are linear transitions, we can calculate any transition directly using $\mathbf{x}(t+\Delta t) = e^{\mathbf{A}\Delta t}\mathbf{x}(t)$.

$$\mathbf{F} := \begin{bmatrix} e^{\mathbf{A}} & 0 \\ 0 & e^{\mathbf{A}} \end{bmatrix} \qquad \mathbf{Q} := \begin{bmatrix} \bar{\mathbf{Q}} & 0 \\ 0 & \bar{\mathbf{Q}} \end{bmatrix} \tag{85}$$

We used $c = 0.06, \tau = 0.17, \Delta t := 1$ and covariance

$$\bar{\mathbf{Q}} := 0.1^2 \cdot \begin{bmatrix} \frac{1}{3} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 3 \end{bmatrix} \tag{86}$$

The matrix exponential is computed using Bader et al. [2019]. The parameters for the emission distribution:

$$\mathbf{H} := \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \qquad \mathbf{R} := 0.5^2 \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{87}$$

We simulate a $K := 131,072$ trajectory for training, $K := 16,384$ trajectory for validation and $K := 32,768$ for testing. The $\tilde{\mathbf{F}}$ that is used in the (non-optimal) Kalman filter and hybrid model is computed as follows:

$$e^{\mathbf{A}\Delta t} \approx \tilde{\mathbf{F}} := \sum_{n=0}^{1} (\Delta t \mathbf{A}^n)/n! \tag{88}$$

G.2 Lotka-Volterra

The Lotka-Volterra equations describe the nonlinear dynamics of a biological system in which the population numbers of two species (predator-prey) interact. It is described by

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} = \begin{bmatrix} \alpha & -\beta x_1 \\ \gamma x_2 & -\delta \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{89}$$

where x_1 is the number of prey and x_2 the number of predators. We use $\alpha := 1$, $\beta := 0.2$, $\gamma := 1$, $\delta := 1$. We integrate this ODE at $dt = 0.001$ and select 4096 evenly-spaced time-steps at $\Delta t = 0.6$. $K := 4096$ are sampled for training, testing and evaluation. $\tilde{\mathbf{F}}_k$ is obtained by a first-order Taylor expansion of the exponential of $\mathbf{A}_{|\mathbf{x}_k}$ evaluated at $\mathbf{x}_k$.

$$e^{\mathbf{A}_{|\mathbf{x}_k}\Delta t} \approx \tilde{\mathbf{F}} := \sum_{n=0}^{1} (\Delta t \mathbf{A}_{|\mathbf{x}_k}^n)/n! \tag{90}$$

We use $\mathbf{H} := \mathbf{I}$ and $\mathbf{R} := 0.5^2\mathbf{I}$.

G.3 Lorenz Equations

We simulate a Lorenz system according to

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} = \begin{bmatrix} -\sigma & \sigma & 0 \\ \rho - x_1 & -1 & 0 \\ x_2 & 0 & -\beta \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}. \tag{91}$$

We integrate the system using $dt = 0.00001$ and sample it uniformly at $\Delta t = 0.05$. We use $\rho = 28$, $\sigma = 10$, $\beta = 8/3$. The transition in Δt arbitrary time-steps is linearly approximated by a Taylor expansion and used in the Kalman smoother and hybrid models.

$$e^{\mathbf{A}_{|\mathbf{x}_k}\Delta t} \approx \tilde{\mathbf{F}}_k := \sum_{n=0}^{2} (\Delta t \mathbf{A}_{|\mathbf{x}_k})^n/n! \tag{92}$$

We simulate $K := 131,072$ steps for training, $K := 32,768$ for testing and $K := 16,384$ for validation. We have $\mathbf{H} := \mathbf{I}$ and thus $\mathbf{x} \in \mathbb{R}^3$ and $\mathbf{y} \in \mathbb{R}^3$. We use $\mathbf{R} := 0.5^2\mathbf{I}$.