Learning to Infer Fast by Attending to Sparse Temporal Observations

Mehrnaz Motamed[1*] **Harry Bendekgey**[2*] **Debora Sujono**[3] **Erik B. Sudderth**[1]

[1]Department of Computer Science, University of California, Irvine
[2]Department of Computer Science, Tufts University
[3]Department of Computer Information Systems, Crafton Hills College
[*]Equal contribution

Abstract

Many scientific datasets contain sparse, irregular observations of dynamic processes. Fast inference techniques are required to infer the unseen evolution of physical systems underlying growing datasets, and estimate the parameters that govern them. We first accelerate the convergence of Gauss-Markov variational inference via efficient natural gradient computation and automatic step-size adaptation. We then propose two general-purpose schemes for initialization from sparse data, one using junction tree representations of Gauss-Markov distributions, and another attention-based neural network architecture that learns to predict correlated temporal posteriors. We show that these initializers dramatically improve performance for ecological models of soil decomposition and predator-prey dynamics. We further extend our methods to a semiparametric epidemiological model of SARS-CoV-2 RNA wastewater measurements, which uses Gaussian processes to model uncertainty in latent reproduction rates. Our methods allow for quick inference of both local states and global dynamical parameters, and unlike prior work, can be applied even when measurements are sparse and the generative process is not fully known.

1 INTRODUCTION

Generative models are an important tool for understanding the world we live in. A scientist taking noisy measurements of soil carbon might be interested in the ground truth concentration over time, as well as identifying the underlying rate of decomposition governing the process. For both of these goals, learning to infer the underlying process quickly is crucial to a scalable scientific method.

We focus on generative models of temporal data whose coarse structure is known, but where some dynamical parameters (such as reaction rates) must be inferred. Given noisy measurements of physical processes, often taken at irregular time intervals, scientists need tools to efficiently infer both the evolution of the underlying processes and the dynamical parameters. A popular technique for this setting is variational inference, which optimizes the fit of an approximate posterior taken from a tractable family, often an exponential family of distributions [Wainwright and Jordan, 2008] such as Gaussians [Kucukelbir et al., 2017]. But if we restrict ourselves to factorized (uncorrelated) Gaussians, we will systematically underestimate uncertainty [Giordano et al., 2018]. If we consider all Gaussian distributions with arbitrary (dense) covariance, the number of parameters grows quadratically with data length, slowing computation and inducing local optima [Sujono et al., 2022].

We expand a body of literature that resolves these issues by encoding the Markov temporal structure of the true posterior in our variational Gaussian. Unlike prior work [Archer et al., 2015, Bamler and Mandt, 2017, Ryder et al., 2018, 2021], we focus on realistic problem settings where observations are taken at irregular time steps and may be sparse. We are the first to apply natural gradients [Amari, 1998] broadly to this problem setting, leveraging recent advances in efficient and automatic natural gradient computation [Bendekgey et al., 2023]. Our key novel contributions are in developing schemes to initialize optimization, a design choice that can dramatically speed convergence and avoid local optima, but is not emphasized by prior variational inference methods.

Our first novel initialization exploits junction tree representations [Cowell et al., 1999] of covariances to initialize posterior dependencies across time while incorporating sparse observations in a principled way. Our second initialization uses an attention-based [Vaswani, 2017] neural network architecture to *learn* how to initialize Gauss-Markov distributions. Conventional amortized inference networks, as developed for *variational autoencoders* (VAEs) [Kingma and Welling, 2014, Mnih and Gregor, 2014, Rezende et al., 2014], assume the full generative model is fixed. Our initial-

ization routines instead take the global dynamics parameters as input, allowing them to work in diverse regimes at test time. Rather than naively processing parameters with neural network layers, our inference network draws samples from the generative prior, and uses attention layers to choose among the possible trajectories given sparse observation queries. Natural gradient refinement of these strong initializations quickly produces accurate posteriors.

After defining our modeling assumptions (Sec. 2) and discussing related work (Sec. 3), we develop methods for efficiently computing natural gradients and adapting learning rates in Sec. 4, and our novel initialization methods in Sec. 5. We show that these techniques effectively uncover the inner workings of ecological phenomena from sparse, noisy observations in Sec. 6. Generalizing the state space models assumed by most related work, we also extend our variational inference methods to a semiparametric epidemiological model of SARS-CoV-2 RNA wastewater measurements in Los Angeles, which uses non-Markov Gaussian processes to model uncertainty in latent reproduction rates.

2 GAUSS-MARKOV VARIATIONAL INFERENCE & STATE SPACE MODELS

A *stochastic differential equation* (SDE, Särkkä and Solin [2019]) models the evolution of a latent state x over time t,

$$dx_t = \alpha(x_t, \theta, t)\, dt + \beta(x_t, \theta, t)^{1/2}\, dW_t, \qquad (1)$$

where x_t is a D-dim. vector of state variables, $\alpha(x_t, \theta, t)$ is a D-dim. *drift vector*, $\beta(x_t, \theta, t)$ is a $D \times D$ *diffusion matrix*, and W_t is the standard Wiener (Brownian motion) process. The dynamics depend on parameters $\theta \in \mathbb{R}^K$. For example, for the Lotka-Volterra model of predator-prey dynamics [Renshaw, 1991] illustrated in Fig. 1, the two latent state variables ($D = 2$) model the prey and predator populations. Dynamical parameters $\theta \in \mathbb{R}^3$ specify rates of prey reproduction, predation, and predator death. Detailed equations for $\alpha(x_t, \theta, t)$ and $\beta(x_t, \theta, t)$ are in the appendix.

We are given sparse, noisy observations y_t of the state x:

$$p(y_t \mid x_t) = \mathcal{F}(y_t; \mu(x_t), \sigma(x_t)). \qquad (2)$$

Here, $\mathcal{F}$ is some family of distributions whose mean $\mu(x_t)$ and standard deviation $\sigma(x_t)$ depend on the latent state x_t. $\mathcal{F}$ could be normal, or a heavier-tailed distribution that gives robustness to the outliers that are prevalent in many scientific datasets. This generative process is an example of a non-linear *state space model* [Anderson and Moore, 1979, Särkkä and Svensson, 2023]: given the latent state x_t, the past and future of the process are independent of y_t.

Given observations $y = (y_{t_1}, y_{t_2}, \dots)$ which may be sparse and taken irregularly over time as in Fig. 1, our goal is to infer the posterior of the latent state sequence. We temporally discretize the SDE and learn a posterior $p(x \mid y, \theta)$ on a finite number of steps $x = (x_1, \dots, x_T) \in \mathbb{R}^{TD}$. Sometimes we seek to infer latent trajectories x given known dynamics parameters θ, and sometimes we jointly infer both x and θ. In both cases, optimization of the local posterior $q(x; \lambda)$ must be fast and stable, since optimization of $q(\theta; \nu)$ requires good (nearly-converged) estimates of state posteriors.

2.1 VARIATIONAL INFERENCE

Variational inference [Jordan et al., 1999, Wainwright and Jordan, 2008, Blei et al., 2017] determines an approximation $q(x; \lambda)$ of the true posterior distribution $p(x \mid y, \theta)$ by maximizing the *evidence lower bound* (ELBO):

$$\mathcal{L}(\lambda) = \mathbb{E}_{q(x;\lambda)}[\log p(x, y \mid \theta) - \log q(x; \lambda)]. \qquad (3)$$

Maximizing the ELBO is equivalent to minimizing the KL divergence $D(q(x; \lambda) \mid\mid p(x \mid y, \theta))$ with respect to λ.

Whereas classic work [Winn and Bishop, 2005, Beal and Ghahramani, 2006] employed carefully handcrafted families of distributions $q(x; \lambda)$ to enable closed-form coordinate ascent optimization, *automatic differentiation variational inference* (ADVI) instead approximates all posteriors with Gaussians [Kucukelbir et al., 2017]. For our Gauss-Markov posteriors, the entropy $\mathbb{E}_{q(x;\lambda)}[-\log q(x; \lambda)]$ can be computed in closed form (see appendix). Unbiased stochastic estimates of the expected log-prior and log-likelihood may be produced via Monte Carlo sampling from $q(x; \lambda)$.

When x is constrained, we let $x = f(\tilde{x})$ and optimize a Gaussian posterior on $\tilde{x}$ given a constraint-enforcing function f, such as `exp` or `softplus` for positive x. This change of variables requires the addition of a Jacobian log-determinant to the loss. For notational simplicity, we use $q(x; \lambda)$ to denote our variational posterior in later sections.

Inferring θ. In some cases, learning about x is not the primary goal, and we instead hope to infer the generative process θ. Given a global prior $p(\theta)$ and sequences of observations y generated by θ, we optimize an expanded ELBO:

$$\mathcal{L}(\lambda, \nu) = \mathbb{E}_{q(x;\lambda)q(\theta;\nu)}[\log p(x, y, \theta) - \log q(x; \lambda)q(\theta; \nu)] \qquad (4)$$

Because θ is typically low-dimensional, we parameterize the global parameters' posterior $q(\theta; \nu)$ via a Gaussian with a full-rank covariance. We optimize Eq. (4) by *coordinate ascent*: at each iteration, we draw a batch of observation sequences y as in stochastic VI [Hoffman et al., 2013], and optimize $q(x; \lambda)$ to obtain λ^*. We then update $q(\theta; \nu)$ with the natural gradient $\tilde{\nabla}_\nu \mathcal{L}(\lambda^*, \nu)$ (see section 2.3). Each update to $q(\theta; \nu)$ requires a (nearly) converged estimate of local state posteriors $q(x; \lambda)$, so inference must be fast.

Amortized Variational Inference. Because stochastic gradient descent may converge slowly, some work instead trains a neural network to output approximate posteriors. If trained

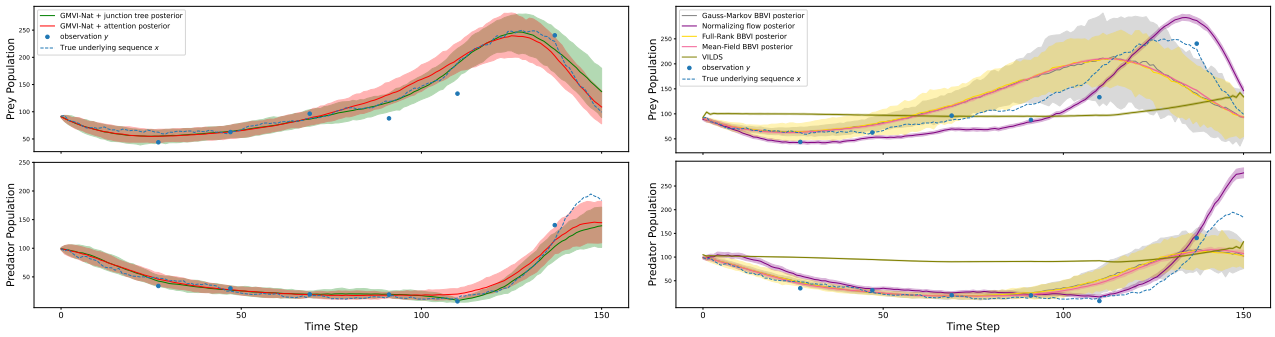

Figure 1: Posterior distributions (after 100 optimization steps) for the Lotka-Volterra model of prey (top) and predator (bottom) dynamics given sparse temporal observations (dots). Solid lines are posterior means, and shaded regions cover two standard deviations. Gauss-Markov correlations between neighboring time steps are inferred but not visualized. Variants of our GMVI, given initializations from Sec. 5, give accurate posteriors (left). BBVI variants (Mean-Field, Full-Rank [Kucukelbir et al., 2017], Gauss-Markov [Bamler and Mandt, 2017]), normalizing flow [Ryder et al., 2021], and VILDS [Archer et al., 2015] baselines (right) poorly estimate uncertainty and fail to cover the true state sequence (dashed).

on a large and representative dataset, *amortized inference encoders* may produce good approximate posteriors [Mnih and Gregor, 2014, Rezende et al., 2014, Kingma and Welling, 2014]. However, our experiments and prior work show the *amortization gap* [Cremer et al., 2018, Krishnan et al., 2018] from optimal inference may be substantial.

2.2 GAUSS-MARKOV VARIATIONAL INFERENCE

Gaussian variational posteriors are commonly parameterized via $\lambda = (\mu, \Sigma)$, where for state space models the mean $\mu \in \mathbb{R}^{TD}$ and covariance $\Sigma \in \mathbb{R}^{TD \times TD}$. The covariance Σ is typically forced to be diagonal (*mean field* inference) or allowed to be a general positive definite matrix (*full-rank* inference) [Kucukelbir et al., 2017]. Mean field approximations neglect (typically strong) temporal correlations in the true posterior, and are known to systematically underestimate uncertainty [Giordano et al., 2018]. Full-rank inference optimizes $O(T^2D^2)$ parameters, which is prohibitively expensive for long sequences and can infer spurious correlations that lead to poor local optima [Sujono et al., 2022].

An alternative explored in some prior work is to define a Gaussian posterior that shares the Markov structure of the generating process. Under the SDE of Eq. (1), $x_{t+\epsilon}$ and $x_{t-\epsilon}$ are independent given x_t. *Gauss-Markov variational inference* (GMVI) seeks a Gaussian that shares this structure, and only infers parameters that directly couple neighboring times. The inverse-covariance or *precision* matrix

$$\Lambda = \Sigma^{-1} = \begin{bmatrix} \Lambda_{11} & \Lambda_{12} & \Lambda_{13} & \dots \\ \Lambda_{21} & \Lambda_{22} & \Lambda_{23} & \dots \\ \Lambda_{31} & \Lambda_{32} & \Lambda_{33} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}, \tag{5}$$

where each Λ_{st} is a $D \times D$ sub-block that defines the conditional precisions between x_s and x_t. When two components s, t of x are conditionally independent given other components, we have $\Lambda_{st} = 0$ [Speed and Kiiveri, 1986, Cowell et al., 1999, Wainwright and Jordan, 2008]. For our Gaussian posterior to inherit the Markov structure of the prior, we set $\Lambda_{st} = 0$ for $|s-t| > 1$, inducing a block tri-diagonal matrix. The resulting Λ has $O(TD^2)$ parameters, and we optimize it in place of the covariance matrix. While the precision matrix Λ is sparse, the induced covariance Σ captures correlations between states separated by arbitrary time lags.

2.3 NATURAL GRADIENT PRECONDITIONING

Variational inference relies on (stochastic) gradient descent to optimize the ELBO. Standard gradient descent implicitly uses Euclidean distance as its notion of distance between parameter vectors, which is a poor indicator of the divergence between distributions and may slow convergence. *Natural gradients*, denoted by $\widetilde{\nabla}$, instead implicitly assume a KL-divergence metric between parameters [Amari, 1998] by rescaling the gradient by the inverse Fisher information matrix $F_\lambda = \mathbb{E}_{q(x;\lambda)}[-\nabla^2_\lambda \log q(x;\lambda)]$ of the posterior:

$$\widetilde{\nabla}_\lambda \mathcal{L}(\lambda) = F_\lambda^{-1} \nabla_\lambda \mathcal{L}(\lambda). \tag{6}$$

For exponential families including Gaussians, we can avoid explicitly computing F_λ by employing the *natural parameterization* $\lambda = (\eta_1, \eta_2) = (\Lambda\mu, -\frac{1}{2}\Lambda)$. This parameterization leads to cancellations in Eq. (6), yielding a simple natural gradient that can be computed via automatic differentiation [Malagò et al., 2011, Khan and Nielsen, 2018]. However, natural parameters have constraints ($-2\eta_2$ must be positive definite) that naive updates may violate. For conditionally conjugate models, step sizes may be easily selected to maintain constraints [Hoffman et al., 2013], but general state-space models lack such guarantees. In Sec. 4.1, we show how modern automatic differentiation tools may nevertheless be used to optimize Gauss-Markov natural parameters while guaranteeing that Λ stays positive definite.

3 RELATED WORK

Some prior work has used block tri-diagonal precision matrices Λ for variational inference. Bamler and Mandt [2017] assume the state dimension $D = 1$, and propose a scheme for optimizing the posterior mean μ and the *Cholesky decomposition* R, $\Lambda = RR^T$, of the precision. Samples may be drawn via the reparameterization $x = \mu + R^{-T}z$, $z \sim \mathcal{N}(0, I_T)$, allowing backpropagation of ELBO gradients to λ. For this scalar Gauss-Markov posterior, they also show how (standard, *not* natural) gradients may be efficiently computed via a temporal recursion similar to the classic Kalman smoother [Anderson and Moore, 1979].

Archer et al. [2015] uses this parameterization for amortized inference. Their most accurate architecture parameterizes the posterior Gaussian by multiplying Gaussian approximations of the prior and likelihood. The prior approximation is restricted to be zero-mean, linear, and time-invariant.

A non-archival workshop paper by Sujono et al. [2022] considers temporally sparse observations, and implicitly defines $q(x; \lambda)$ via a linear dynamical system, training (via standard gradients) time-varying transition parameters to adapt to observations. Fortuin et al. [2020] use Gauss-Markov parameterizations for time series imputation via a model integrating VAEs with Gaussian processes.

Ryder et al. [2018] propose recurrent neural networks, and Ryder et al. [2021] propose normalizing flows, for unamortized variational inference in state-space models. These models allow posteriors to be non-Gaussian, but fail to capture the Markov structure of the true posterior. Their networks are trained on subsequences and applied to a longer sequence, amortizing across time but requiring computationally expensive retraining for each new observation sequence.

Our goals are distinct from latent SDE and neural ODE models that assume the differential equation dynamics and/or observation model are complex functions that need to be parameterized by a neural network, with inference possibly occurring in an unidentifiable latent space [Hasan et al., 2021, Rubanova et al., 2019, Tzen and Raginsky, 2019]. These papers require inference neural networks to approximate their deep generative processes. We instead consider scientifically motivated, interpretable generative models and seek to accurately model a small but crucial set of dynamics parameters. Scientists need accurate uncertainty measures for parameter estimates, motivating our approach in Sec. 5.

4 ACCELERATING GAUSS-MARKOV VI

Efficient natural gradient computation requires appropriate parameterization of the variational posterior. Prior GMVI methods [Archer et al., 2015, Bamler and Mandt, 2017] optimized the mean μ and Cholesky factor R of Λ. We instead take $\lambda = (\eta_1, R)$, where $\eta_1 = \Lambda\mu$ is the first natural parameter. Using R enforces the positive-definiteness constraint that directly optimizing $\eta_2 = -\frac{1}{2}\Lambda$ may violate. This choice also eases initialization: while initializing μ requires state estimates at *all* time steps, η_1 defines a conditional distribution at each time [Cowell et al., 1999], avoiding the imputation heuristics required by prior work.

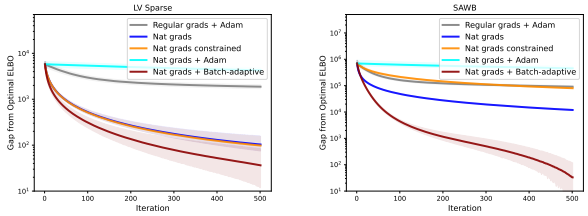

Figure 2: Comparison of gradient-based optimizers for the ELBO of Eq. (3) for two SDE models (see Sec. 6). We initialize from a temporally-factorized $q(x; \lambda)$ that matches the marginal mean and variance of the prior $p(x \mid \theta)$, and plot mean and 95% confidence intervals across sequences, versus iteration. For each model and method, we set the learning rate γ to the largest value that avoids instability. For natural gradients with Adam, descent is only possible for tiny learning rates, and optimization stagnates.

4.1 AUTODIFF NATURAL GRADIENTS

For training structured VAE models with latent representations defined by graphical models, Bendekgey et al. [2023] introduced a method for computing natural gradients as in Eq. (6) via automatic differentiation on unconstrained reparameterizations of λ, guaranteeing that constraints are satisfied after updates. We extend this approach to GMVI by optimizing the unconstrained Cholesky factor R rather than the constrained parameter $-\frac{1}{2}\Lambda$. We efficiently compute $\widetilde{\nabla}_\lambda \mathcal{L}(\lambda)$ via a straight-through estimator [Bengio et al., 2013] and PyTorch's forward-mode autodiff.

In the forward pass, (η_1, R) are mapped to moment parameters $(\mu, \Sigma + \mu\mu^\top)$ before computing the ELBO. Unlike the low-dimensional setting of Bendekgey et al. [2023], our application requires inverting the $TD \times TD$ precision matrix Λ. Exploiting its Markov structure, we compute μ and $\Sigma = \Lambda^{-1}$ using an efficient belief propagation algorithm [Wainwright and Jordan, 2008] related to the Kalman smoother [Särkkä and Svensson, 2023]. As detailed in App. B, performing these steps outside autodiff avoids memory blow-up and compounding numerical errors. Belief propagation implicitly ensures that μ is temporally consistent and informed by all observations.

Fig. 2 shows that natural gradients converge faster than standard gradients, but only when using the unconstrained Cholesky parameterization. Directly optimizing the constrained precision Λ (orange curve) requires a much smaller learning rate to maintain positive-definiteness, making per-

formance similar to standard gradients with Adam [Kingma and Ba, 2015]. Other autodiff natural gradients have assumed conjugacy [Hoffman et al., 2013], required specialized loss modifications [Lin et al., 2020], or computed biased gradients [Johnson et al., 2016]. Instead, we efficiently compute unbiased natural gradients of the true ELBO.

4.2 BATCH-ADAPTIVE NATURAL GRADIENTS

The Adam optimizer [Kingma and Ba, 2015] updates each parameter $\lambda_{b\ell}$ using $\lambda_{b\ell} \leftarrow \lambda_{b\ell} - \gamma \frac{\hat{m}_{b\ell}}{\sqrt{\hat{v}_{b\ell}+\epsilon}}$, where $\hat{m}_{b\ell}$ and $\hat{v}_{b\ell}$ are exponentially weighted estimates of the first and second gradient moments for batch sequence $b \in \{1, \ldots, B\}$ and parameter $\ell \in \{1, \ldots, L\}$ of λ, and γ is the learning rate. Applied to natural gradients, Adam performs poorly: by rescaling each component independently, it reverts to Euclidean geometry and distorts the natural-gradient manifold.

To preserve geometry while retaining adaptivity, we normalize at the sequence level. For each sequence b, $\lambda_{b\ell} \leftarrow \lambda_{b\ell} - \gamma \frac{\hat{m}_{b\ell}}{\sqrt{\hat{v}_b+\epsilon}}$, where $\hat{v}_b \leftarrow \beta \hat{v}_b + (1-\beta)\|g_b\|^2$ is a running estimate of the squared ℓ_2-norm of the natural gradient $g = \tilde{\nabla}_\lambda \mathcal{L}$, shared across parameters within the sequence. This *batch-adaptive* normalization allows different sequences to have different effective learning rates, improving the rate of natural gradient descent in practice (Fig. 2).

Prior work has shared normalizers across parameters for memory efficiency in neural network training [Zhang et al., 2024], and adapted learning rates for natural gradient descent in simpler conditionally conjugate models [Ranganath et al., 2013]. Neither approach enables per-sequence adaptation of natural gradient learning rates, as we do.

5 INITS WITH TEMPORAL STRUCTURE

Efficient gradient-based inference requires good initialization of the approximate posterior $q(x; \lambda)$. While most prior work has relied on handcrafted (often undocumented) initialization heuristics, we propose two complementary and general-purpose approaches for automatic initialization of Gauss-Markov posteriors. The first is a principled but simple junction-tree method that approximates the prior and likelihood with Gaussians, and initializes $q(x; \lambda)$ to their product. The second is a flexible attention-based neural network that is trained to output correlated, Gauss-Markov posteriors. The attention-based initializer (Sec. 5.2) makes integral use of junction tree representations, and most improves on direct junction tree initialization (Sec. 5.1) when the non-Gaussian generative prior has multiple distinct modes.

For both initializers, we initialize once and then refine $q(x; \lambda)$ with the natural gradient updates of Sec. 4. When jointly inferring $q(x; \lambda)$ and $q(\theta; \nu)$, we re-initialize $q(x; \lambda)$ after each update of θ before refining with natural gradients.

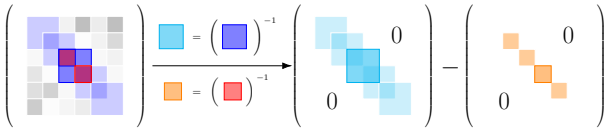

Figure 3: Junction-tree algorithm for fast precision initialization, visualized. We compute the correlations between features at each time step (red) and between pairs of neighboring time steps (purple), ignoring the other $O(T^2D^2)$ possible correlations (gray). By inverting these $D \times D$ and $2D \times 2D$ blocks, adding them together in the regions in which they overlap (cyan), and subtracting a correction for overlap (orange), we obtain the sparse Λ.

5.1 INITIALIZATION VIA JUNCTION TREES

We approximate the prior $p(x \mid \theta)$ with a Gauss-Markov distribution $\hat{p}(x; \theta)$, the likelihood $p(y \mid x)$ with a Gaussian $\hat{p}(x; y)$ in x, and then initialize the variational posterior $q(x; \lambda) \propto \hat{p}(x; \theta)\hat{p}(x; y)$. To compute the product of two Gaussians, we simply add their natural parameters. This technique elegantly handles missing observations by setting the natural parameters to 0 at time steps without data. The mean μ is then implicitly defined by the output of a Kalman smoother on these natural parameters, unlike prior work requiring interpolation heuristics to initialize μ (see Sec. 3).

For many scientific applications like the ones we consider in Sec. 6, the likelihood function $p(y \mid x)$ of Eq. (2) is easily approximated via a Gaussian; see App. A.4. Producing a Gaussian approximation of the prior is harder because it does not factorize across time. We can compute an approximation by drawing samples and computing the empirical mean and covariance, but the empirical distribution will not exactly satisfy the assumed Markov properties; in general its precision matrix is dense. If we want to produce a block tri-diagonal (and therefore Gauss-Markov) posterior initialization $q(x; \lambda)$, we need our approximate prior $\hat{p}(x; \theta)$ to be Gauss-Markov. If we are overly restrictive and force our prior approximator $\hat{p}(x; \theta)$ to factorize independently across time, our resulting initial $q(x; \lambda)$ will also factorize, an unrealistic restriction that starts us far from the optimum (see the poor Gauss-Markov BBVI starting point in Fig. 7).

After drawing M samples, rather than computing all T^2D^2 empirical covariances, we compute the covariances *only between neighboring time steps*. We then rely on a key exponential family property: there exists exactly one Gauss-Markov distribution which has the sufficient statistics (mean and positive definite local covariances) we've computed. By applying the junction tree decomposition [Cowell et al., 1999], we can find that unique Λ, and initialize a Gauss-Markov distribution whose local properties match our prior. More precisely, the junction tree algorithm maps the block tri-diagonal entries of the covariance matrix (the covariance is *not* assumed to be sparse, other entries are generally

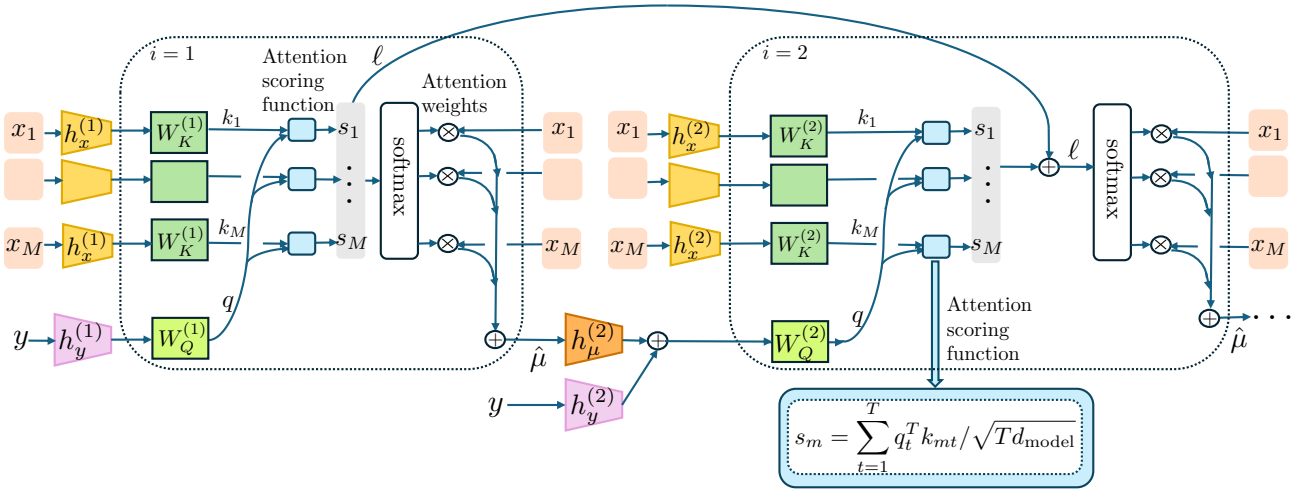

Figure 4: Attention architecture for predicting the posterior mean given sparse observations and M samples from the generative prior, applied in parallel to a batch of B sequences y. Here, $h_\mu^{(i)}, h_y^{(i)}, h_x^{(i)}$ are multi-layer perceptrons (MLPs), and $W_Q^{(i)}, W_k^{(i)}$ are learnable query and key matrices in layer i. We set $h_y^{(i)}(y) = 0$ for time steps with no observations.

nonzero) to the unique matrix with block tri-diagonal sparsity structure, whose inverse matches the observed local moments. This mapping is unique whenever the observed covariances may be embedded inside a positive definite matrix, which happens with probability one if enough samples are drawn to produce strictly positive definite (not just positive semi-definite) local moments.

Fig. 3 visualizes this approach intuitively: we invert each overlapping diagonal $2D \times 2D$ block and add them, subtracting the inverses of the diagonal $D \times D$ blocks. These inverses can be quickly computed in parallel, unlike general matrix inversion algorithms which are iterative. From general exponential family theory [Wainwright and Jordan, 2008], this initialization can also be interpreted as a maximum likelihood estimate of the Gauss-Markov parameters given the M samples from the prior $p(x \mid \theta)$. See App. C for a proof and more detailed algorithm specification.

In Fig. 7 we see this initialization dramatically outperforms baselines, but for more sparse observations, there is a gap between its performance and that of our learned attention-based initializer (see Sec. 5.2). To understand why, consider a prior $p(x \mid \theta)$ that has multiple modes, but the true posterior $p(x \mid y, \theta)$ is unimodal due to the high confidence we draw from observations. In this case, the general GMVI restriction that $q(x; \lambda)$ be Gaussian comes at little cost, but constructing it by first approximating the prior with a Gaussian poorly captures the true posterior covariance.

5.2 ATTENTION ARCHITECTURES FOR GMVI

We now propose an attention-based neural network architecture which *learns* initializations that: (1) like the junction-tree method, adapt to the current value of θ and directly accounts for the prior, but (2) uses more than just the first and second moments of $p(x \mid \theta)$, and (3) can handle sparse observations. Unlike standard *encoders* that implicitly adapt their weights to a fixed generative model, our network must be able to take θ as input at *test-time* within a coordinate ascent loop. While we could simply make θ an input to a conventional encoder, the mapping from low-dimensional θ to high-dimensional $p(x \mid \theta)$ may be hard to learn from a modest training set. Our solution leverages the fact that this mapping is already defined via the generative model.

We design our architecture to therefore take in *samples* from $p(x \mid \theta)$, and use irregular observations to output a weighted combination of prior samples, effectively learning to choose between prior modes. This naturally fits the attention framework [Vaswani, 2017]: our network takes B observation sequences (*queries*) and M prior sample sequences (*keys*), and outputs a weighted sum of those prior samples (*values*). Unlike standard transformers, which use positional encodings to capture sequential data, permutation invariance is desirable here as both prior samples and the batch lack order.

Inferring the posterior mean. Our initialization network first estimates the posterior mean $\hat{\mu} = \sum_{m=1}^{M} w_m x_m$ as a convex combination of M prior sample trajectories $x_m \in \mathbb{R}^{TD}, m = 1, \ldots, M$, given parameters θ. Intuitively, informative data leads to posteriors with less uncertainty than the prior, so that mixtures of prior trajectories span plausible posterior means. Any residual bias is corrected by the natural-gradient refinement of Sec. 4.

The attention architecture of Fig. 4 refines $\hat{\mu}$ over n layers with residual connections. Each layer contains an attention module and multilayer perceptrons (MLPs) $\{h_\mu^{(i)}, h_y^{(i)}, h_x^{(i)}\}_{i=1}^{n}$, which operate independently across time and output sequences of size $T \times d_{\text{model}}$. Queries are

Algorithm 1 Attention-Based Encoder Architecture

Require: M prior samples $x \sim p(x \mid \theta) : M \times T \times D$
Require: Observation sequence $y : T \times D$
$\ell \leftarrow 0, \quad \hat{\mu} \leftarrow \texttt{None}, \quad h_\mu^{(1)}(\hat{\mu}) \leftarrow 0$
for $i \leftarrow 1$ to n **do**
  $q \leftarrow W_Q^{(i)} \left(h_\mu^{(i)}(\hat{\mu}) + h_y^{(i)}(y) \right)$
  for $m \leftarrow 1$ to M **do**
    $k_m \leftarrow W_K^{(i)} h_x^{(i)}(x_m)$
    $s_m \leftarrow \left(\sum_{t=1}^T q_t^T k_{mt} \right) / \sqrt{T d_{\text{model}}}$
  $\ell \leftarrow \ell + s$
  $w \leftarrow \texttt{softmax}(\ell)$
  $\hat{\mu} \leftarrow \sum_m w_m x_m$
return $\hat{\mu}$

Figure 5: Algorithm pseudocode for estimating the posterior mean, performed in parallel for a batch of B (sparse) sequences y. See Fig. 4 for the architecture and notation.

formed from observations and the running mean estimate $\hat{\mu}$, while keys/values are transforms of prior samples x_m. The attention score $s_m = \frac{1}{\sqrt{T d_{\text{model}}}} \sum_{t=1}^T q_t^\top k_{mt}$ measures the similarity between queries and keys across time. Log-weights ℓ_m accumulate across layers ($\ell \leftarrow \ell + s$), and weights are calculated with $w \leftarrow \texttt{softmax}(\ell)$, progressively refining $\hat{\mu}$. When batching multiple observation sequences, all sequences can share the same M prior samples. We set $h_y^{(i)}(y) = 0$ at unobserved times in each batch, so the attention sum can be computed given sparse observations.

Inferring the posterior precision. Given $\hat{\mu}$, we obtain the posterior precision through a variation of Alg. 1 detailed in App. D. Instead of observations y, we use the mean-prediction network output $\hat{\mu}$ as the query. In addition to accruing a modified mean estimate $\tilde{\mu}$, we return the weights $\tilde{w}_m$ from the final attention layer and compute

$$\Sigma_{t:t+1} = \sum_{m=1}^M \tilde{w}_m x_{m,t:t+1} x_{m,t:t+1}^T - \tilde{\mu}_{t:t+1} \tilde{\mu}_{t:t+1}^T, \quad (7)$$

an attention-weighted empirical covariance of the prior samples. We then construct the posterior by adapting the junction tree transform of Sec. 5.1: we define Λ by inverting the covariances of Eq. (7), and add observation precisions to all time steps with observations. This structured covariance mapping improves the stability of amortized training.

6 EXPERIMENTS

We evaluate GMVI inference using synthetic data from Lotka-Volterra models of predator-prey evolution over time [Renshaw, 1991, Boys et al., 2008] and stochastic Allison-Wallenstein-Bradford (SAWB) models of soil biogeochemistry [Allison et al., 2010]. We further evaluate our approach on the epidemiological EIRR-ww model applied to real SARS-CoV-2 (the causative agent of COVID-19) RNA measurements from Los Angeles wastewater [Goldstein et al., 2024]. Exact formulations of $p(x, y, \theta)$, and experiments on a simpler near-linear soil biogeochemical model, are in App. A.3-A.4. Experimental details including network architectures and hyperparameters are in App. A.5.

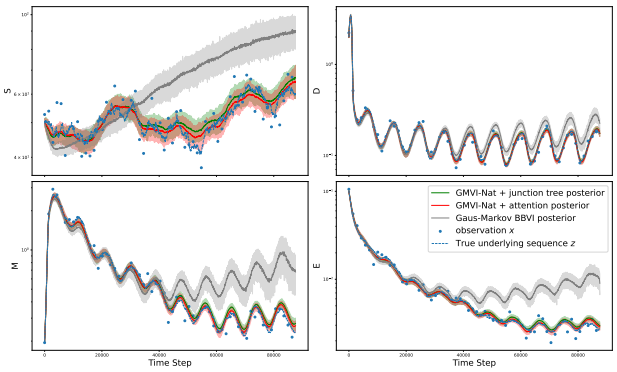

Figure 6: Visualization of inferred Gauss-Markov posteriors for the SAWB model given sparse data. Solid lines are posterior means and shaded regions cover 2 standard deviations. The Gauss-Markov BBVI baseline [Bamler and Mandt, 2017] fails to capture the true generating sequence.

The Lotka-Volterra model exhibits nonlinear dynamics and multimodality due to the interplay between predator and prey populations. We study time series of length $T = 150$, with Gaussian observations either densely sampled at all time points (`LVDense`), or with waiting times drawn from a Poisson distribution with mean 20 (`LVSparse`).

The influential SAWB soil biogeochemical model simulates the evolution of carbon in soil as the climate around it changes. As the climate warms, soil decomposes carbon more quickly and releases CO_2, resulting in a positive feedback loop that accelerates rising global temperatures. We consider long time series of 10 years, discretized at a resolution of 2 days, yielding $T = 1825$. We assume observations are taken every 60 days for 61 total observations (see Fig. 6).

For the EIRR-ww model, we primarily seek to infer the time-varying transmission rate (r_t, also known as the *effective reproduction rate*) of COVID-19 in Los Angeles county using wastewater genome concentration data. Conditioned on r_t, a latent dynamical system describes the evolution of infected, infectious, and recovering individuals between 07/04/2021 and 02/23/2022. The data are sparsely sampled and irregular in time, with up to three measurement replicates per sampling day. To allow for high noise levels, the likelihood is a heavy-tailed Student's t distribution.

Our EIRR generative model is inspired by Goldstein et al. [2024] but has some differences. First, we replace their latent ODE dynamics of infected and infectious individuals with an SDE whose drift matches, but contains added noise

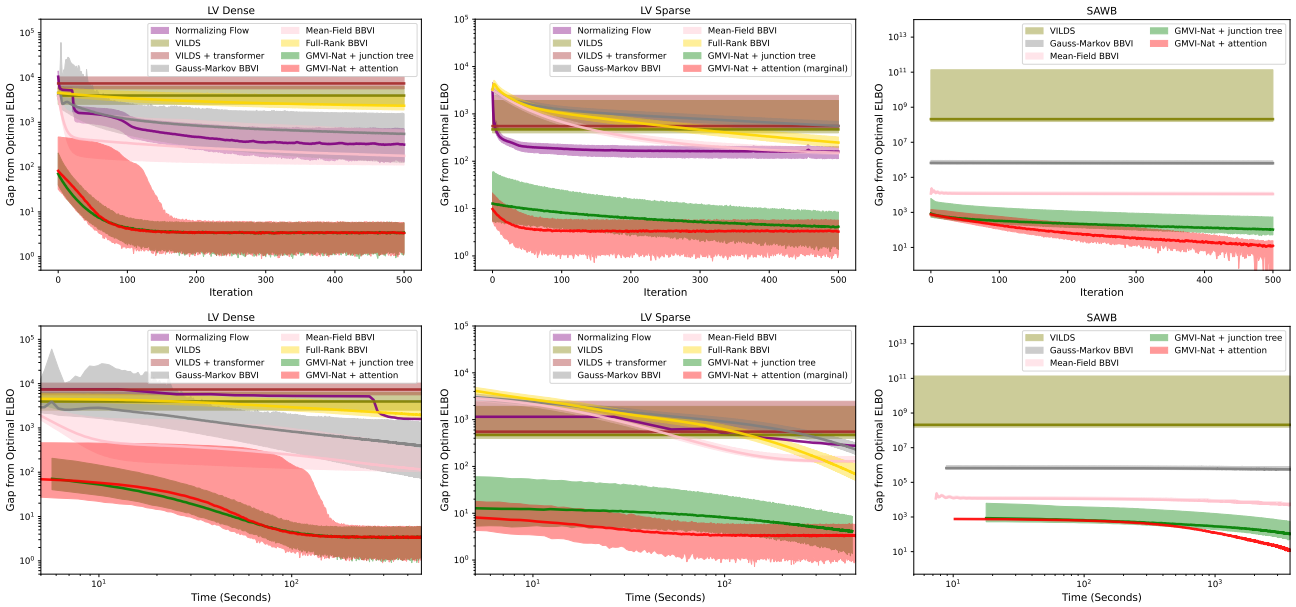

Figure 7: Loss trajectory versus inference iteration (top, linear scaling) and versus wall clock time (bottom, log scaling). Results are aggregated across 1024 sequences (for Lotka-Volterra) and 64 sequences (for SAWB), with shaded regions covering 95% confidence intervals and solid lines marking the mean across sequences. The optimal ELBO is the best final ELBO observed among all runs. VILDS [Archer et al., 2015] is an amortized method which does not improve over time, whose inference network is tested with both dense layers (as in the original paper) and transformers. Mean-Field BBVI accuracy suffers from inaccurate independence assumptions, while Full-Rank BBVI [Kucukelbir et al., 2017] and Gauss-Markov BBVI [Bamler and Mandt, 2017] and normalizing flows [Ryder et al., 2021] struggle to converge from the weaker initializations assumed in prior work. (Normalizing flow also requires a problem-dependent, non-amortized "pre-training" phase whose cost is not plotted; its true speed is slower.) No Full-Rank BBVI result is shown for the longer SAWB test sequences because it runs out of memory, demonstrating the importance of modeling the true Markov posterior.

to allow for model misspecification. Second, their simple random-walk prior model for r_t only allowed rates to vary on a weekly basis; we instead use a nonparameteric Gaussian process [Williams and Rasmussen, 2006] whose Matérn covariance kernel allows for short-term transmission surges or "waves", and enables us to infer r_t at daily resolution. This semiparametric model is non-Markov, but our variational inference framework still allows us to infer an accurate Gauss-Markov posterior approximation, with changes required only to evaluate the GP defining $\log p(x \mid \theta)$.

Inferring x. In Fig. 2 we compare optimization methods to investigate the impact of natural gradients and batch-adaptive normalization on descent. All methods are initialized with a posterior that factorizes across time, and matches the mean and variance of the prior at each time. We denote our best method, natural gradient optimization with batch-adaptive normalization, as `GMVI-Nat` for the remaining experiments. We initialize `GMVI-Nat` in two ways (Sec. 5).

In Fig. 7, we compare `GMVI-Nat` with our novel junction tree and attention network initializations to baselines. We first consider the unamortized (standard) gradient optimization of Bamler and Mandt [2017], which shares the independent-across-time initialization with Fig. 2 (`Gauss-Markov BBVI`). As their hand-derived gradient recursion is specialized to simple models with scalar states ($D = 1$) and dense observations, we instead compute gradients via automatic differentiation. We also compare to variational inference with fully factorized (`Mean-Field BBVI`) and unconstrained (`Full-Rank BBVI`) Gaussian posteriors, optimized via automatic differentiation (standard) gradients [Kucukelbir et al., 2017]. Finally, we consider the unamortized, non-Markov normalizing flow method of Ryder et al. [2021] (`Normalizing Flow`).

Our amortized baseline `VILDS` is based on an inference network architecture that smooths predicted means over time. While Archer et al. [2015] assumed dense observations, we extend to allow sparse observations. We also train a variant of VILDS that replaces fully-connected network layers with transformers, to illustrate major differences from the novel way in which our initialization network uses attention. Our initialization network and `VILDS` are trained to optimize Eq. (3) on training sequences before testing on distinct data.

Our initialization network could also be trained in a semi-amortized fashion [Kim et al., 2018], where the loss *after (natural) gradient refinement* is computed at training time. However, we see little empirical improvement from this

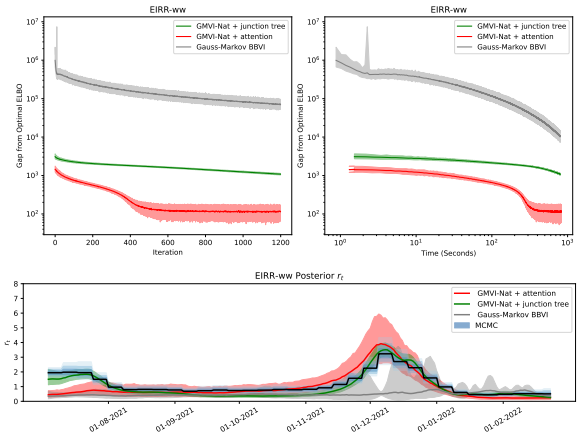

Figure 8: *Top:* Loss trajectory versus training iteration (left) and wall-clock time (right, log scaling). Confidence intervals and the optimal ELBO are determined via multiple inference trials as in Fig. 7. *Bottom:* Comparison of posterior reproduction number $r(t)$ estimates produced by GMVI and an MCMC baseline. The MCMC approach requires 72 seconds for MAP initialization, and sampling takes 7,620 seconds in total, for a simpler model that assumes r_t is constant within weeks (unlike our higher-resolution GP model). Further details and log-probability traces are in App. A.5.

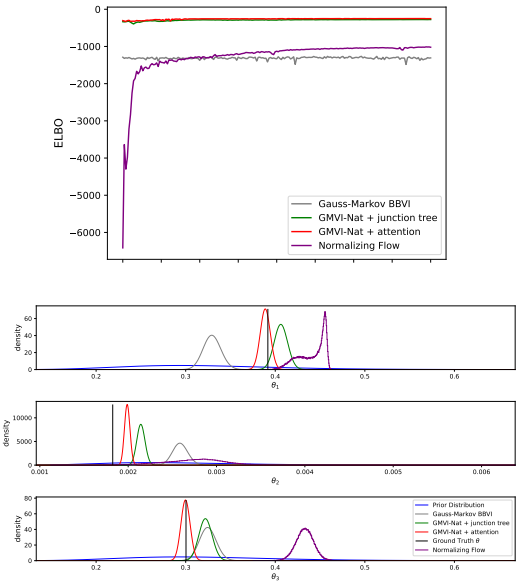

Figure 9: Dynamical parameter θ inference given sparse Lotka-Volterra observations. For each coordinate ascent method, we perform 100 optimization iterations to estimate $q(x;\lambda)$ followed by a natural gradient step to improve $q(\theta)$. The normalizing flow method [Ryder et al., 2021] optimizes $q(x;\lambda), q(\theta;\nu)$ in parallel. We show the ELBO value (see Eq. (4)) against outer optimization iteration (top), and the learned posteriors after 200 updates to $q(\theta;\nu)$ (bottom).

memory-intensive technique (see App. A.1), and repeated inner-optimization dramatically slows training.

Fig. 7 shows the loss trajectory of VI methods over inference iteration and time. Both of our initializations produce consistently high-quality approximate posteriors, from which natural gradient descent quickly converges to state estimates that dramatically improve on baselines (see Fig. 1, 6).

Fig. 8 shows loss trajectories for the EIRR-ww model of SARS-CoV-2 data over inference iteration and time, along with a comparison of the inferred posterior reproduction number to the MCMC baseline. Both initialization strategies rapidly converge to high-quality approximate posteriors, with the attention estimate leading to better uncertainty estimates and a superior ELBO. Differences between our inferred posterior, and the MCMC estimate from prior work, seem primarily to be due to differences in modeling assumptions. Our GMVI produces approximate posteriors far faster than the prior MCMC approach, taking only a few minutes instead of multiple hours, in spite of incorporating a more sophisticated Gaussian process prior.

Inferring θ. In Fig. 9, we show how these inference methods can be used to infer the dynamics parameters θ. Lotka-Volterra has 3 parameters: a prey growth rate θ_1, a predation rate θ_2, and a predator death rate θ_3. Given 128 sequences which share the same unknown θ, we perform coordinate ascent on Eq. (4) to produce the visualized $q(\theta;\lambda)$. Our novel inference techniques give a strong signal for updating θ after only 100 inner optimization steps, while baselines are too far from convergence to give high-quality estimates.

7 DISCUSSION

By learning strong initializations and refining via quick, stable natural gradient steps, we more efficiently infer variational state and parameter posteriors for dynamical models. Our approach flexibly handles sparse or dense observations, allowing us to learn the underlying dynamics of the world around us. Our quick convergence in a diverse set of regimes may enable future applications to large-scale scientific data.

Acknowledgements

This research was supported in part by NSF Robust Intelligence Award No. IIS-1816365, ONR Award No. N00014-23-1-2712, and the HPI Research Center in Machine Learning and Data Science at UC Irvine. We thank I. H. Goldstein and V. M. Minin for providing the SARS-CoV-2 wastewater RNA data, and sharing insights about their EIRR epidemiological model. We thank H. W. Xie and S. D. Allison for determining realistic generative parameters for the SAWB and SCON soil biogeochemical models.

References

Steven D Allison, Matthew D Wallenstein, and Mark A Bradford. Soil-carbon response to warming dependent on microbial physiology. *Nature Geoscience*, 3(5):336–340, 2010.

Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.

B. D. O. Anderson and J. B. Moore. *Optimal Filtering*. Prentice Hall, New Jersey, 1979.

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.

Richard Bamler and Stephan Mandt. Structured Black Box Variational Inference for Latent Time Series Models. In *ICML Time Series Workshop*, 2017.

Matthew J. Beal and Zoubin Ghahramani. Variational Bayesian learning of directed graphical models with hidden variables. *Bayesian Analysis*, 1(4):793 – 831, 2006.

Henry C Bendekgey, Gabe Hope, and Erik Sudderth. Unbiased learning of deep generative models with structured discrete representations. *Advances in Neural Information Processing Systems*, 36:69849–69886, 2023.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

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, 2017.

Richard J Boys, Darren J Wilkinson, and Thomas BL Kirkwood. Bayesian inference for a discretely observed stochastic kinetic model. *Statistics and Computing*, 18: 125–135, 2008.

R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer-Verlag, New York, 1999.

Chris Cremer, Xuechen Li, and David Duvenaud. Inference suboptimality in variational autoencoders. In *International conference on machine learning*, pages 1078–1086. PMLR, 2018.

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

Ryan Giordano, Tamara Broderick, and Michael I Jordan. Covariances, robustness, and variational bayes. *Journal of machine learning research*, 19(51):1–49, 2018.

Isaac H. Goldstein, Daniel M. Parker, Sunny Jiang, and Volodymyr M. Minin. Semiparametric inference of effective reproduction number dynamics from wastewater pathogen surveillance data. *Biometrics*, 80(3): ujae074, 08 2024. ISSN 0006-341X. doi: 10.1093/biomtc/ujae074. URL `https://doi.org/10.1093/biomtc/ujae074`.

Ali Hasan, Joao M Pereira, Sina Farsiu, and Vahid Tarokh. Identifying latent stochastic differential equations. *IEEE Transactions on Signal Processing*, 70:89–104, 2021.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 2013.

Matthew J Johnson, David K Duvenaud, Alex Wiltschko, Ryan P Adams, and Sandeep R Datta. Composing graphical models with neural networks for structured representations and fast inference. *Advances in neural information processing systems*, 29, 2016.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2): 183–233, 1999.

Mohammad Emtiyaz Khan and Didrik Nielsen. Fast yet simple natural-gradient descent for variational inference in complex models. In *2018 International Symposium on Information Theory and Its Applications (ISITA)*, pages 31–35. IEEE, 2018.

Yoon Kim, Sam Wiseman, Andrew Miller, David Sontag, and Alexander Rush. Semi-amortized variational autoencoders. In *International Conference on Machine Learning*, pages 2678–2687. PMLR, 2018.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

Rahul Krishnan, Dawen Liang, and Matthew Hoffman. On the challenges of learning with inference networks on sparse, high-dimensional data. In *AISTATS*, pages 143–151, 2018.

Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M Blei. Automatic differentiation variational inference. *Journal of machine learning research*, 18(14):1–45, 2017.

S. L. Lauritzen. *Graphical Models*. Oxford University Press, Oxford, 1996.

Wu Lin, Mark Schmidt, and Mohammad Emtiyaz Khan. Handling the positive-definite constraint in the bayesian learning rule. In *International conference on machine learning*, pages 6116–6126. PMLR, 2020.

Luigi Malagò, Matteo Matteucci, and Giovanni Pistone. Towards the geometry of estimation of distribution algorithms based on the exponential family. In *Proceedings of the 11th workshop proceedings on Foundations of genetic algorithms*, pages 230–242, 2011.

Andriy Mnih and Karol Gregor. Neural variational inference and learning in belief networks. In *International Conference on Machine Learning*, pages 1791–1799, 2014.

R. Ranganath, C. Wang, D. M. Blei, and E. P. Xing. An adaptive learning rate for stochastic variational inference. In *International Conference on Machine Learning*, pages 298–306, 2013.

Eric Renshaw. *Modelling biological populations in space and time*. Cambridge University Press, 1991.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, 2014.

Yulia Rubanova, Ricky TQ Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32, 2019.

Thomas Ryder, Dennis Prangle, Andrew Golightly, and Isaac Matthews. The neural moving average model for scalable variational inference of state space models. In *Uncertainty in Artificial Intelligence*, pages 12–22. PMLR, 2021.

Tom Ryder, Andrew Golightly, A Stephen McGough, and Dennis Prangle. Black-box variational inference for stochastic differential equations. In *International Conference on Machine Learning*, pages 4423–4432. PMLR, 2018.

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

Simo Särkkä and Lennart Svensson. *Bayesian filtering and smoothing*, volume 17. Cambridge university press, 2023.

Terence P Speed and Harri T Kiiveri. Gaussian markov distributions over finite graphs. *The Annals of Statistics*, pages 138–150, 1986.

Debora Sujono, Hua W Xie, Steven Allison, and Erik B Sudderth. Variational inference for soil biogeochemical models. In *AI for Science Workshop at the International Conference on Machine Learning*, 2022.

Belinda Tzen and Maxim Raginsky. Neural stochastic differential equations: Deep latent gaussian models in the diffusion limit. *arXiv preprint arXiv:1905.09883*, 2019.

D Ulyanov. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1:1–305, 2008.

Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

J. Winn and C. M. Bishop. Variational message passing. *Journal of Machine Learning Research*, 6:661–694, 2005.

Yushun Zhang, Congliang Chen, Ziniu Li, Tian Ding, Chenwei Wu, Yinyu Ye, Zhi-Quan Luo, and Ruoyu Sun. Adammini: Use fewer learning rates to gain more. In *Workshop on Efficient Systems for Foundation Models II@ ICML2024*, 2024.

Learning to Infer Fast by Attending to Sparse Temporal Observations (Supplementary Material)

Mehrnaz Motamed[1*] **Harry Bendekgey**[2*] **Debora Sujono**[3] **Erik B. Sudderth**[1]

[1]Department of Computer Science, University of California, Irvine
[2]Department of Computer Science, Tufts University
[3]Department of Computer Information Systems, Crafton Hills College
[*]Equal contribution

A EXPERIMENTAL DETAILS AND SUPPLEMENTARY RESULTS

In this section we provide supplementary results and provide of our training and testing regimes. Full code to replicate experiments will be provided.

A.1 COMPARISON TO SEMI-AMORTIZED TRAINING

We train our neural network architecture to output high-quality posteriors, then measure the quality of those posteriors after natural gradient refinement. A logical follow-up question, therefore, is whether performance can be improved by training specifically for the task being evaluated at test time. Instead of training to optimize the loss at the output, we could take refinement steps *at training time too*, and perform gradient descent on the ELBO of the refined posterior.

This method of *semi-amortized variational inference* [Kim et al., 2018] is very costly, as inner refinement iterations must be performed for every gradient step of neural network training. To keep training time low, we performed 10 optimization steps and compared the performance of the resulting neural network to the one trained to optimize its output (amortized). We saw virtually no difference between the performances, so we demonstrated results in the main paper on the computationally-cheaper amortized training regime. See Fig. 10 for an example comparison.

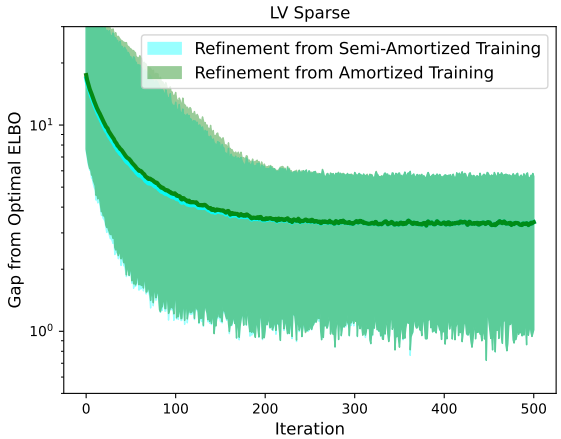

Figure 10: Comparison of refinement loss after training fully amortized vs semi-amortized. Intervals almost perfectly overlap; shared fluctuations are due to a shared random seed.

A.2 RESULTS ON LOTKA-VOLTERRA AND SAWB MODELS

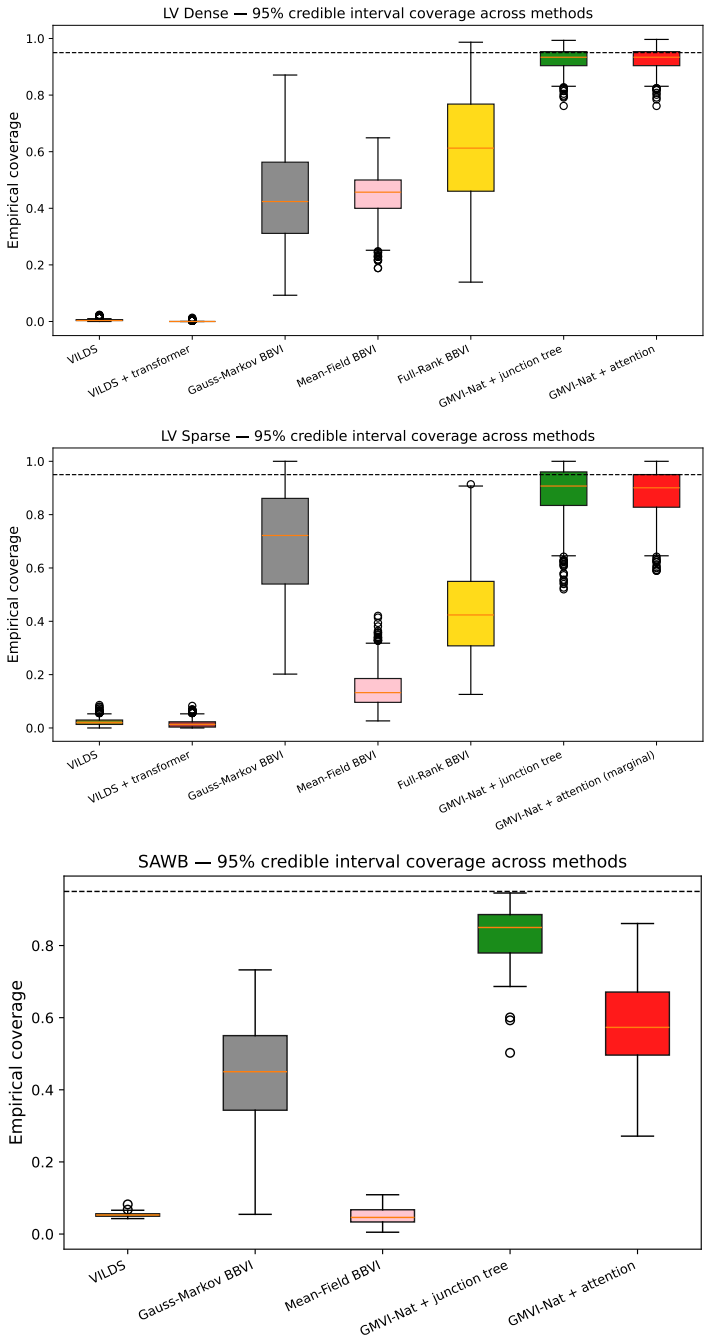

Figure 11: Empirical coverage of 95% credible intervals for Lotka-Volterra model in dense (top) and sparse (middle) settings and SAWB model (Bottom), across inference methods after 500 iterations. For each method and test sequence, we draw 200 posterior samples from the posterior distribution $q(x; \lambda)$ and compute the empirical 2.5th and 97.5th percentiles at each coordinate of time-steps and state-dimensions (T, D). We then compute the fraction of coordinates of the ground-truth trajectory x^* that fall within this interval, yielding one coverage value per test sequence. Boxplots summarize the distribution of this per-sequence coverage across all $B = 1024$ test sequences for each method; the dashed line marks the nominal target of 0.95. A well-calibrated posterior should place its coverage distribution near this line: intervals that are systematically too narrow (overconfident) fall below it, as with VILDS and Mean-Field BBVI.

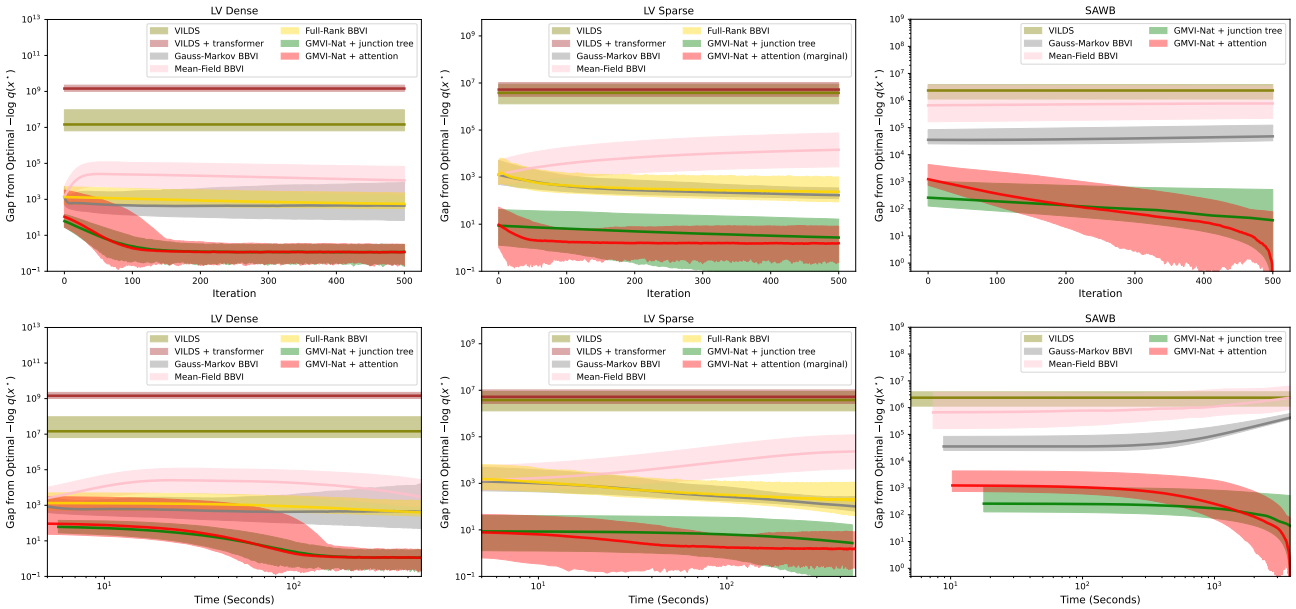

Figure 12: Negative log-likelihood (NLL) gap versus inference iteration (top, linear scale) and versus wall-clock time (bottom, log scale), where the gap is the shortfall between the variational posterior's log density at the ground-truth latent trajectory x^* and the best final negative log-density observed among all runs. This shows how close the learned posterior actually comes to the true latent trajectory. Results are aggregated across 1024 sequences (Lotka–Volterra) and 64 sequences (SAWB), with shaded regions showing 95% confidence intervals and solid lines marking the mean across sequences.

A.3 RESULTS ON SCON MODEL

Here, we provide results on another generative model with near-linear Gaussian dynamics and dense observations. The stochastic linear conventional (SCON) soil biogeochemical model is specified in Sec. A.4, and is linear except for the diffusion turn and observation uncertainty scaling with x_t.

This regime exposes a weakness of our attention architecture: it only takes in the observations indirectly, using them to choose between prior samples. Thus, when samples are dense and informative, it may be better to "connect the dots" instead of choosing between prior samples to find one that matches. This is particularly true when the prior $p(x)$ is Gaussian or near-Gaussian, because the posterior will be fully unimodal and the Gaussian approximation in our junction tree initialization will be accurate.

Results can be seen in Fig. 14. Even in this regime, Archer et al. [2015] is unable to do more than slight smoothing over a connect-the-dots solutions.

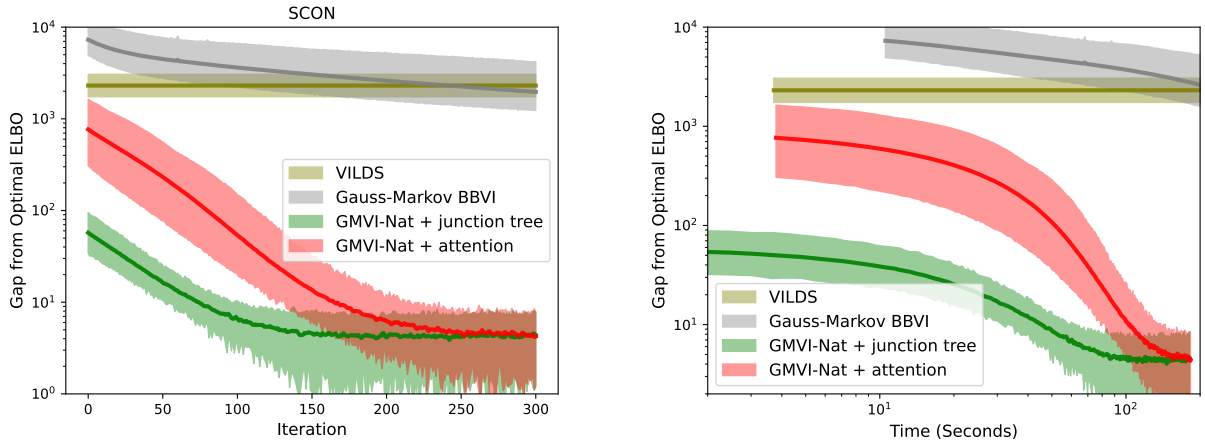

Figure 13: Gap from optimal ELBO versus inference iteration (left, linear scale) and versus wall-clock time (right, log scale), on the SCON model with near-linear dynamics. Both panels use a log scale on the y-axis. Shaded regions show 95% confidence intervals and solid lines mark the mean across test sequences. Both GMVI-Nat variants converge to a substantially smaller gap than VILDS and Gauss-Markov BBVI.

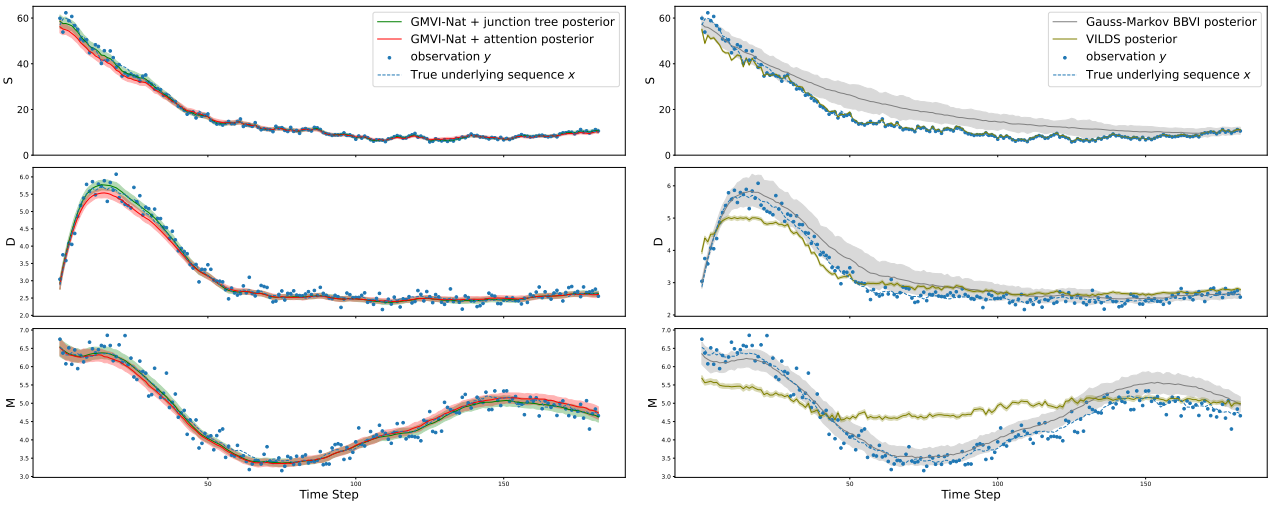

Figure 14: Posterior mean trajectories (solid lines) and 95% credible intervals (shaded bands) for all three latent dimensions of the SCON model (S, D, M; rows) after 100 inference steps, for a single sequence. Left: both GMVI-Nat variants (junction tree and attention) closely track the true latent trajectory x (dashed) and remain tightly concentrated around it. Right: Gauss-Markov BBVI shows visibly wider credible intervals and VILDS learns a posterior mean that fails to track the true trajectory.

A.4 GENERATIVE MODEL SPECIFICATIONS

All the models below are described by SDEs of the form

$$dx_t = \alpha(x_t, t, \theta)dt + \beta(x_t, \theta, t)dW_t. \tag{8}$$

All models except for EIRR-ww use a simple Gaussian observation model:

$$p(y_t \mid x_t) = \mathcal{N}(y_t; \mu = Ax_t, \sigma = Cx_t). \tag{9}$$

Thus we set $f(x_t) = Ax_t, g(x_t) = Cx_t$. The natural choice for likelihood-approximating functions are $\tilde{f}(y;\theta) = A^{-1}y$ and precision matrix $\tilde{g}(y;\theta) = (A^T(Cy)^{-1}A)^{-1}$ where the inner inverse is performed element-wise.

Thus to define each model, we need to define **(i)** the dimensions of x_t, **(ii)** the parameters θ and corresponding drift and diffusion terms $\alpha(x_t, t, \theta), \beta(x_t, \theta, t)$, **(iii)** the observation model parameters A, C, **(iv)** the total time interval and discretization resolution dt, **(v)** the scheme for how observation times are determined, and **(vi)** the priors on global parameters and initial time step $p(\theta), p(x_0)$.

Lotka-Volterra. We have state $x_t = \begin{bmatrix} u_t & v_t \end{bmatrix}^T$, where u_t is the number of prey and v_t is the number of predators alive at time t. Given prey reproduction rate θ_1, predation rate θ_2, and predator death rate θ_3, we get drift and diffusion terms:

$$\alpha(x_t, \theta) = \begin{bmatrix} \theta_1 u_t - \theta_2 u_t v_t \\ \theta_2 u_t v_t - \theta_3 v_t \end{bmatrix} \tag{10}$$

And diffusion given by

$$\beta(x_t, \theta) = \begin{bmatrix} \theta_1 u_t + \theta_2 u_t v_t & -\theta_2 u_t v_t \\ -\theta_2 u_t v_t & \theta_2 u_t v_t + \theta_3 v_t \end{bmatrix} \tag{11}$$

We set $A = I, C = 0.2I$. To remove discretization artifacts, we replace all normal distributions (i.e. in $p(x_{t+dt}|x_{dt})$ and $p(y_t|x_t)$) with transformed normal distributions for both sampling and loss evaluation to ensure positivity of $x, y \sim p(x, y|\theta)$. We use the same transformation to constrain our posterior, $x = \texttt{softplus}(\tilde{x} - 1) + 1$ (to ensure the populations don't get unrealistically close to) and we fit a Gaussian variational factor to $\tilde{x}$.

We set the maximum time $T = 30$ and $dt = 0.2$, yielding 151 time steps. We consider 2 different regimes for observation times: one in which an observation is taken at each time point, and one in which observation waiting time is drawn from a Poisson distribution with mean 20.

The prior and initial state distributions are given by

$$\begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} \sim \text{softplus}\,\mathcal{N}\left(\mu = \begin{bmatrix} -1 \\ -6 \\ -1 \end{bmatrix}, \sigma = \begin{bmatrix} \sqrt{0.1} \\ \sqrt{0.1} \\ \sqrt{0.1} \end{bmatrix}\right) \tag{12}$$

$$\begin{bmatrix} u_0 \\ v_0 \end{bmatrix} \sim \mathcal{N}\left(\mu = \begin{bmatrix} 91 \\ 99 \end{bmatrix}, \sigma = \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right) \tag{13}$$

For reference, the mean values of $\theta_1, \theta_2, \theta_3$ are $\approx 0.3, 0.002$, and 0.3.

SAWB. The SAWB model [Allison et al., 2010] has a latent dimension of 4, with dimensions of the latent variables $x_t = \begin{bmatrix} S_T & D_t & M_t & E_t \end{bmatrix}^T$ corresponding to soil organic carbon (S), dissolved organic carbon (D) and microbial biomass (M), and enzyme concentration (E). The drift vector is given by:

$$\begin{bmatrix} I_S + a_{MSA} \cdot r_M \cdot M - \frac{V_{DE} \cdot E \cdot S}{K_{DE} + E + S} \\ I_D + (1 - a_{MSA}) \cdot r_M \cdot M + \frac{V_{DE} \cdot E \cdot S}{K_{DE} + E + S} + r_L \cdot E - \frac{V_{UE} \cdot M \cdot D}{K_{UE} + M + D} \\ u_Q \cdot \frac{V_{UE} \cdot M \cdot D}{K_{UE} + M + D} - (r_M + r_E) \cdot M \\ r_E \cdot M - r_L \cdot E \end{bmatrix} \tag{14}$$

Values in red are functions of time, but have had their dependencies dropped to simplify notation. I_S, I_D are independent of θ:

$$I_S(t) = I_D(t) = 0.001 + 0.0005 \cdot \sin(2\pi t/24/365) \tag{15}$$

and the remaining terms are functions of underlying parameters

$$V_i = V_{i,\text{ref}} \exp\left\{ -\frac{E a_{V_i}}{R}\left(\frac{1}{T} - \frac{1}{T_{ref}}\right)\right\} \tag{16}$$

where R is the ideal gas constant $8.314\ J \cdot K^{-1} \cdot \text{mol}^{-1}$ and T_{ref} specifies a "reference equilibrium temperature which we set to 283 K. The current temperature T is given by the idealized equation

$$T = T_{\text{ref}} + \frac{T_{\text{rise}} \cdot t}{80 \cdot 24 \cdot 365} + 10\sin(2\pi t/24) + 10\sin(2\pi t/24/365). \tag{17}$$

Param	Distribution	Truncation
$u_{Q,\text{ref}}$	$\mathcal{N}(0.2, 0.1)$	$[0, 1]$
Q	$\mathcal{N}(2 \cdot 10^{-3}, 10^{-3})$	$[0, 1]$
a_{MSA}	$\mathcal{N}(0.5, 0.25)$	$[0, 1]$
K_{DE}	$\mathcal{N}(500, 500)$	$[0, 10^4]$
K_{UE}	$\mathcal{N}(1, 1)$	$[0, 10^2]$
$V_{DE,\text{ref}}$	$\mathcal{N}(0.5, 0.25)$	$[0, 10]$
$V_{UE,\text{ref}}$	$\mathcal{N}(0.01, 0.005)$	$[0, 1]$
$Ea_{V_{DE}}$	$\mathcal{N}(50, 25)$	$[0, 150]$
$Ea_{V_{UE}}$	$\mathcal{N}(60, 30)$	$[0, 150]$
r_M	$\mathcal{N}(2 \cdot 10^{-3}, 10^{-3})$	$[0, 1]$
r_E	$\mathcal{N}(5 \cdot 10^{-6}, 2.5 \cdot 10^{-6})$	$[0, 1]$
r_L	$\mathcal{N}(5 \cdot 10^{-4}, 2.5 \cdot 10^{-4})$	$[0, 1]$
s_S	$\mathcal{N}(10^{-3}, 10^{-4})$	$[0, 1]$
s_D	$\mathcal{N}(10^{-3}, 10^{-4})$	$[0, 1]$
s_M	$\mathcal{N}(10^{-3}, 10^{-4})$	$[0, 1]$
s_E	$\mathcal{N}(10^{-3}, 10^{-4})$	$[0, 1]$

Table 1: Priors on SAWB global parameters θ.

Because time is measured in hours, this accounts for daily and yearly cycles in temperature, and we use the high estimate of $T_{\text{rise}} = 5$ Celsius temperature rise globally by 2100. The final time-varying parameter is given by:

$$u_Q = u_{Q,\text{ref}} + Q \cdot (T - T_{\text{ref}}). \tag{18}$$

We note the much simpler diffusion matrix:

$$\beta(x_t, \theta) = \begin{bmatrix} s_S \cdot S & 0 & 0 & 0 \\ 0 & s_D \cdot D & 0 & 0 \\ 0 & 0 & s_M \cdot M & 0 \\ 0 & 0 & 0 & s_E \cdot E \end{bmatrix} \tag{19}$$

and observation model $A = I, C = 0.1I$. We assume observations are taken regularly every 2 days, $dt = 48$ and the entire observation sequence lasts 10 years, $T = 10 \cdot 365 \cdot 24$ for 1825 total time steps in each sequence.

We set a prior distribution $p(x_0) = \mathcal{N}(x_0; \mu = \mu_0, \sigma = 0.01\mu_0$ with $\mu_0 = \begin{bmatrix} 50 & 2 & 0.2 & 0.1 \end{bmatrix}^T$. Our priors on our global parameters are given in Table 1. Priors were implemented with truncated normal distributions.

As in the Lotka Volterra model, we replace all normal distributions with transformed normal distributions to ensure positivity when sampling from $p(x, y|\theta)$ or evaluating the log probability. To avoid warping when values get very small, as is possible in SAWB, we use a scaled softplus $f(x) = \texttt{softplus}(1000 \cdot x)/1000$ as our transformation.

SCON. The stochastic conventional soil biogeochemical modeluses one fewer latent dimension than SAWB $x_t = \begin{bmatrix} S_t & D_t & M_t \end{bmatrix}^T$, and has drift given by

$$\begin{bmatrix} -k_S & a_{DS} \cdot k_D & a_M \cdot k_M \cdot a_{MSC} \\ a_{SD} \cdot k_S & -u_M - k_D & a_M \cdot k_M \cdot (1 - a_{MSC}) \\ 0 & u_M & -k_M \end{bmatrix} \cdot x_t + \begin{bmatrix} I_S \\ I_D \\ 0 \end{bmatrix}. \tag{20}$$

The values of I_S, I_D are given by Eq. (15) and the values of k_i obey the Arrhenius temperature dependence equation Eq. (16). SCON uses the same observation model and diffusion matrix as SAWB, except for the last latent and observation dimension E which is removed. The resulting model is only non-linear-Gaussian because of the dependence of the diffusion matrix and observation noise on x_t.

Param	Distribution	Truncation
u_M	$\mathcal{N}(2 \cdot 10^{-3}, 10^{-3})$	$[0, 1]$
a_{DS}	$\mathcal{N}(0.5, 0.25)$	$[0, 1]$
a_{SD}	$\mathcal{N}(0.5, 0.25)$	$[0, 1]$
a_M	$\mathcal{N}(0.5, 0.25)$	$[0, 1]$
a_{MSC}	$\mathcal{N}(0.5, 0.25)$	$[0, 1]$
$K_{S,\text{ref}}$	$\mathcal{N}(5 \cdot 10^{-7}, 2.5 \cdot 10^{-7})$	$[0, 1]$
$K_{D,\text{ref}}$	$\mathcal{N}(10^{-3}, 5 \cdot 10^{-4})$	$[0, 1]$
$K_{M,\text{ref}}$	$\mathcal{N}(2 \cdot 10^{-4}, 10^{-4})$	$[0, 1]$
Ea_S	$\mathcal{N}(60, 30)$	$[0, 100]$
Ea_D	$\mathcal{N}(50, 25)$	$[0, 100]$
Ea_M	$\mathcal{N}(50, 25)$	$[0, 100]$
s_S	$\mathcal{N}(10^{-2}, 10^{-3})$	$[0, 1]$
s_D	$\mathcal{N}(10^{-3}, 10^{-4})$	$[0, 1]$
s_M	$\mathcal{N}(10^{-3}, 10^{-4})$	$[0, 1]$

Table 2: Priors on SCON global parameters θ.

We keep $dt = 48$ but set $T = 365 \cdot 24$ to one year, resulting in a shorter sequence of length 182. We assume there are observations at every time step. We use $p(x_0) = \mathcal{N}(x_0; \mu = \mu_0, \sigma = 0.5\mu_0)$ with $\mu_0 = \begin{bmatrix}56.8 & 2.7 & 6.0\end{bmatrix}^T$. The details of the prior on θ, which are implemented via truncated normal distributions, are detailed in Table 2.

EIRR-ww. We have state $x_t = \begin{bmatrix} r_t & E_t & I_t & R_t^{(1)} & R_t^{(2)} \end{bmatrix}^T$. Here, r_t is the log of the effective reproduction number at time t, i.e. $exp(r_t)$ is the average number of individuals a person infected at time t would subsequently infect, if the conditions of the epidemic at time t remained the same. The remaining 4 components are counts of individuals in various states of infection. E_t is the number of infected but not yet infectious individuals. I_t is the number of infectious individuals. $R_t^{(1)}$ is the number of recovered individuals who are not infectious, but still shed SARS-CoV-2 into the wastewater. $R_t^{(2)}$ is the number of recovered individuals no longer shedding pathogen.

We model r_t with a Gaussian Process with a mean of -1 and a Matérn 3/2 covariance kernel given by

$$k_{\text{Matérn-3/2}}(t, t') = \sigma^2 \left(1 + \frac{\sqrt{3}\,|t - t'|}{\ell}\right) \exp\left(-\frac{\sqrt{3}\,|t - t'|}{\ell}\right),$$

where $\sigma^2 = 0.3$ and $\ell = 14$.

The drift and diffusion terms for the remaining 4 components are:

$$\alpha(x_t, \theta) = \begin{bmatrix} \exp(r_t)\,\nu I_t - \gamma E_t \\ \gamma E_t - \nu I_t \\ \nu I_t - \eta R_t^{(1)} \\ \eta R_t^{(1)} \end{bmatrix} \tag{21}$$

And diffusion given by

$$\beta(x_t, \theta) = \begin{bmatrix} \exp(r_t)\,\nu I_t + \gamma E_t & -\gamma E_t & 0 & 0 \\ -\gamma E_t & \gamma E_t + \nu I_t & -\nu I_t & 0 \\ 0 & -\nu I & \nu I_t + \eta R_t^{(1)} & -\eta R_t^{(1)} \\ 0 & 0 & -\eta R_t^{(1)} & \eta R_t^{(1)} \end{bmatrix} \tag{22}$$

For the observation model, we have

$$p(y_t|x_t) = \text{Generalized-t}\Big(\rho\big(\lambda I_t + (1 - \lambda) R_t^{(1)}\big), \tau^2, df\Big) \tag{23}$$

We fix $\gamma = \frac{1}{4}$, $\nu = \frac{1}{7}$, $\eta = \frac{1}{18}$, $\lambda = 0.997$, $\rho = 0.2$, $\text{df} = 3$, and $\tau = 0.5$.

Method	LVDense lr	LVSparse lr
Gauss-Markov BBVI	1e-2	1e-2
Mean-Field BBVI	1	1
Full-Rank BBVI	1e-3	3e-3
GMVI-Nat + junction tree	1	1e-1
GMVI-Nat + attention	1 (1e-3)	1 (1e-3)
VILDS	(1e-3)	–
Normalizing Flow	1e-3	1e-3

Table 3: Optimization learning rates for the fixed-θ Lotka-Volterra experiments. Training learning rates are shown in parentheses, while learning rates used to optimize $q(x)$ on the test data set are shown without.

Method	LVSparse lr	SAWB lr
Regular grads + Adam	1e-4	1e-3
Nat grads + Adam	1e-5	1e-4
Nat grads	1e-3	1e-4
Nat grads constrained	1e-3	1e-5
Nat grads + Batch-adaptive	2	1e5

Table 4: Optimization learning rates for Fig 2.

The initial state distributions are given by

$$\begin{bmatrix} E_0 \\ I_0 \\ R_{10} \\ R_{20} \end{bmatrix} \sim \mathcal{N}\left(\mu = \begin{bmatrix} 2995 \\ 5990 \\ 11055 \\ 65000 \end{bmatrix}, \sigma = \begin{bmatrix} 0.1 \\ 0.1 \\ 0.1 \\ 0.1 \end{bmatrix}\right) \tag{24}$$

A.5 ARCHITECTURE, HYPERPARAMETER, AND DATA SET SPECIFICATIONS

For all experiments, the random seed required to replicate our results will be included in our code repository.

Lotka-Volterra. The two fixed-θ experiments were done on a batch of 1024 sequences. Amortized methods were trained on a training data set of 8192 sequences which share the same θ as the test set. Both amortized methods were trained on batches of size 128.

To estimate the ELBO, 128 samples from $q(x)$ were drawn. $M = 256$ samples from $p(x)$ were used to compute the junction tree initialization for unamortized methods and perform the prior-sample-selection process of our attention network. Learning rates are included in Table 3. We trained our attention network for 1000 gradient steps and the VILDS baseline for 5000 steps.

For the normalizing flow experiments, we utilized the authors' codebase Ryder et al. [2021], making a few key modifications to improve flexibility in modeling the initial state x_0. In the original implementation, x_0 is determined by a deterministic function. We extend this approach by sampling x_0 according to Equation (24) and modifying the loss function to account for this change, incorporating x_0 into the time series x in Equation (3). Additionally, we apply the transformation $x = \texttt{softplus}(\tilde{x} - 1) + 1$, as detailed in previous sections.

Another modification from the original normalizing flow approach is in how we partition the time series. While the original implementation segments the time series into shorter sequences, we instead set the sequence length equal to the full time series length T, to ensures a more comprehensive temporal representation. Finally, we pre-trained the variational approximation for 500 iterations to obtain reasonable initial values, minimizing $E_{\theta,x\sim q}[||x - \hat{y}||_2]$ where $\hat{y}$ is the linear interpolation of observations.

Lotka-Volterra, infer θ. Global parameter inference was done with a test set of 128 sequences, all drawn from the same θ. These sequences had some sparsity, with observation waiting times drawn from a Poisson distribution with mean 5. Our attention network was trained on a training data set of 1024 *which did not share the same θ as the test set*. The training data

set consisted of batches of size 128, each of which was generated by a different $\theta \sim p(\theta)$. Our network was trained for 1000 iterations with a learning rate of 1e-3.

For optimization, each method took 100 inner optimization steps of $q(x)$ before taking a single natural gradient step to improve $q(\theta)$. GMVI reg used an inner learning rate of 1e-2 and an outer learning rate of 1e-2; GMVI nat + norm used an inner learning rate of 1e-1 and an outer learning rate of 1e-1; and the refinement method GMVI nat + norm + attn used an inner learning rate of 1 and an outer learning rate of 1e-1. To prevent quick descent to a very-tight $q(\theta)$ that would make natural gradient descent slow, the first 50 steps of optimization upweight the KL term $\mathrm{KL}(q(\theta)||p(\theta))$ by the batch size 128. Over the following 50 steps, that factor is annealed down to 1.

For the normalizing flow experiments, we pre-trained for 1000 iteration of simultaneously minimizing $E_{\theta,x\sim q}[||x - \hat{y}||_2]$ where $\hat{y}$ is the linear interpolation of observations and maximizing $E_{x\sim q}[||\theta - \theta^*||_2]$ where $\theta^* = \mathrm{softplus}([-1, -6, -1])$. Both training and pre-training were conducted with a learning rate of 1e-3.

EIRR-ww. To estimate the ELBO, 256 samples from $q(x)$ were drawn. $M = 2048$ samples from $p(x)$ were used to compute the junction tree initialization for unamortized methods and perform the sample-selection process of our attention network. To train the attention network for the EIRR-ww model, instead of sampling from the prior distribution, we sample from the junction tree initial posterior distribution. We trained our attention network for 200 gradient steps.

The baseline MCMC method initializes with MAP (10 restarts) for 122 seconds, and the MCMC sampling with 4 chains takes 7620 seconds. The reproduction number is updated an a weekly basis. Figure 15 depicts the MCMC loss versus iteration.

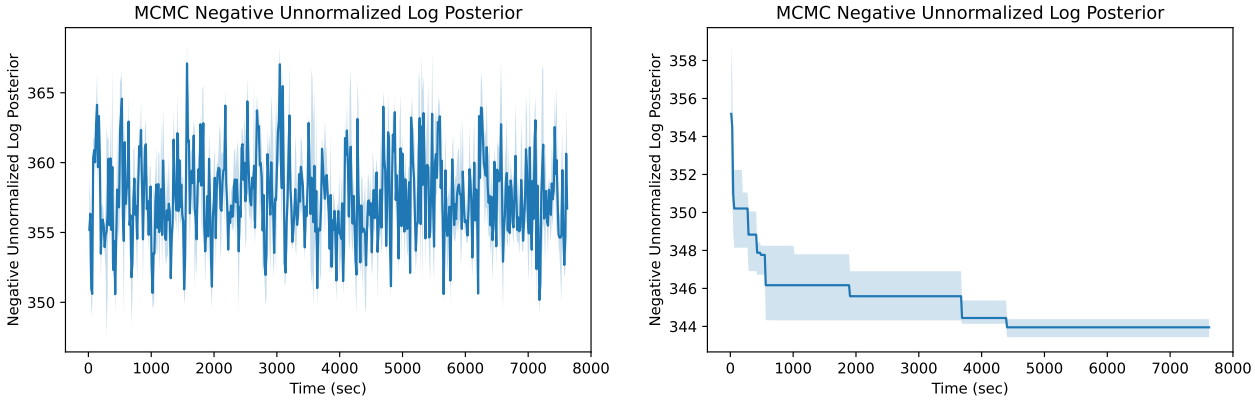

Figure 15: MCMC unnormalized negative log posterior (loss) versus wall clock time (seconds) for the EIRR-ww model on COVID-19 data in Los Angeles county *Left:* At each sampling iteration, we report the median and interquartile range (25th–75th percentile) of the negative unnormalized log posterior (-log joint) across four chains. *Right:* Best-so-far MCMC objective. For each chain, we track the cumulative minimum of the negative unnormalized log posterior (-log joint) over iterations, then summarize across four chains with median and interquartile range.

Comparison of descent methods. Fig. 2 uses the same data and hyperparameters as Fig. 7. The only differences are learning rates (specified in Table 4), and initialization, where all methods inherit the Gauss-Markov BBVI method of independently matches the mean and variance of the prior at each time point and initializing cross-temporal variances to 0.

SAWB. Results are shown on a test data set of 64 sequences. We estimate the ELBO with 128 samples from the prior and we estimate the prior with 256 samples for junction tree initialization. Due to the high memory cost of very long sequences, we lower this value for our amortized training, only drawing 64 samples from the prior. The training data set contained 256 sequences which shared the same θ as the test set, split into batches of 64. Training took 1000 iterations with a learning rate of 1e-3, and $q(x)$ optimization for GMVI reg, GMVI nat + norm, and GMVI nat + norm + attn used learning rates of 1e-3, 5e3, and 1e4 respectively.

SCON. Results are shown on a test data set of 128 sequences. We estimate the ELBO with 128 samples from the prior and we estimate the prior with 256 samples for junction tree initialization. The training data set contained 1024 sequences which shared the same θ as the test set, split into batches of 64. Attention network training took 1000 iterations and VILDS

network training took 10000 iterations, both using a learning rate of 1e-3. $q(x)$ optimization for GMVI reg, GMVI nat, GMVI nat + norm, and GMVI nat + norm + attn used learning rates of 1e-2, 1e-2, 1e2, and 1e2 respectively.

Attention Network architectures. The attention encoder is a standard deep network applied once per sequence; we set $n = 3$ layers for our experiments to balance accuracy and speed (see Alg. 1 in the main paper). The network is trained end-to-end by backpropagating the ELBO of the initialized $q(x; \lambda)$ before refinement. The multilayer perceptrons $u_\mu(\hat{\mu}), u_y(y), k(x)$ were implemented to work independently across time, containing 3 network layers, GELU activation function [Hendrycks and Gimpel, 2016], and instance normalization [Ulyanov, 2016] after every activation. For Lotka-Volterra experiments, we set the hidden dimension to 128 and for the SCON and SAWB experiments we used a hidden dimension of 64.

VILDS architectures. We consider two variants of the VILDS recognition network, differing only in how they extract local observation evidence prior to the closed-form linear-dynamical-system (LDS) update. In both variants, the recognition network consists of three hidden layers of width 128, following an initial linear projection of the observation $y_t \in \mathbb{R}^D$ with GELU activation and instance normalization. The resulting hidden representation is mapped to the variational mean μ and to the likelihood-approximating precision Λ_y, which are combined with the prior-approximating linear dynamical system (whose parameters are learned directly) through closed-form Kalman-smoother-style message passing to yield the full smoothing posterior.

In the dense variant, following the original VILDS formulation of Archer et al. [2015], the remaining two layers are residual fully-connected blocks applied independently at each timestep. Because these are pointwise linear maps, the hidden representation h_t at time t is a function of y_t alone, with no information exchange across timesteps within the network. All temporal dependence is instead introduced downstream via an explicit linear-Gaussian state-space model: a learned transition matrix and process noise covariance combine the per-timestep local evidence potentials produced by the network through closed-form Kalman-smoother-style message passing, yielding the full smoothing posterior.

In the transformer variant, the two residual dense blocks are replaced with standard Transformer encoder layers (four attention heads of dimension 32 each), preceded by a sinusoidal positional encoding to restore the temporal ordering that self-attention otherwise discards. This allows the hidden representation at each timestep to attend to observations at every other timestep in the sequence before local evidence potentials are computed, so that the evidence at time t can incorporate information from temporally distant observations. The downstream combination of local evidence into a full posterior via the linear-Gaussian dynamics is identical to the dense variant; only the feature-extraction front end differs.

The original VILDS recognition network Archer et al. [2015] is designed to condition on a fully dense observation sequence, requiring input at every time step. To apply both VILDS architectures to sparsely-observed data, we first impute the missing time steps for each sequence via per-dimension linear interpolation between the available observations, using linear extrapolation to fill in any values before the first or after the last observed timestep. This dense, interpolated sequence is used solely as the input to the recognition network (for both the MLP-based and transformer-based variants of VILDS). Critically, the interpolated values are never substituted into the model likelihood: the evidence lower bound is still computed using the true sparse observations. This ensures that interpolation serves only to provide a well-formed input to the recognition network, without biasing the variational objective toward interpolated values.

B OPTIMIZATION AND NATURAL GRADIENT SPECIFICATION

In the section that follows, we repeatedly treat Λ as a natural parameter of $q(z)$. Technically, $-\frac{1}{2}\Lambda$ is a natural parameter. To reduce notational bloat, we drop the occasional multiplication by $-\frac{1}{2}$ whenever the natural parameter is constructed or -2 whenever it is converted into an unconstrained or moment parameter.

The training algorithm benefits from having a few functions defined in advance:

- `uncToPrec` takes a lower triangular matrix of unconstrained values and constructs a positive definite precision matrix by: (a) softplusing the values along the diagonal to create a valid Cholesky factor, (b) multiplying that Cholesky factor by its transpose to construct a positive definite matrix, and (c) adding 10^{-6} to the diagonal to ensure numerical stability.
- `precToUnc` is the inverse: it computes the cholesky of a positive definite matrix and then applies the inverse of the softplus transform to its diagonal.
- `forwardProp` takes a function, an input to that function (primal value), and a tensor of perturbations (the dual value)

Algorithm 2
Inferring $q(\theta)$

Require: Observation data $y_b, b = 1, \ldots, B$, step size α, number of optimization steps T, prior parameters μ_0, Σ_0 of $p(\theta)$,
$\eta_\theta \leftarrow \Sigma_0^{-1}\mu_0$
$L_\theta \leftarrow \texttt{precToUnc}(\Sigma_0^{-1}/100)$
for $t \leftarrow 1$ to T **do**
  $\Lambda_\theta \leftarrow \texttt{uncToPrec}(L_\theta)$
  $\mu_\theta \leftarrow \texttt{requiresGrad}(\Lambda_\theta^{-1}\eta_\theta)$
  $S_\theta \leftarrow \texttt{requiresGrad}(\Lambda_\theta^{-1} + \mu_\theta\mu_\theta^T)$
  $\Sigma_\theta \leftarrow S_\theta - \mu_\theta\mu_\theta^T$
  for $b \leftarrow 1$ to B **do**
    $\theta_b \leftarrow \texttt{sampleUsingMoments}(\mu_\theta, S_\theta)$
    $\eta_x, L_x \leftarrow \texttt{inferQX}(y_b, \texttt{detach}(\theta_b))$
    $\ell_b \leftarrow \texttt{calcNELBO}(\eta_x, L_x, \theta_b)$
  $\mathcal{L} \leftarrow KL(p(\theta; \mu_0, \Sigma_0) \| q(\theta; \mu_\theta, \Sigma_\theta)) + \sum_{b=1}^{B} \ell_b$
  $\eta_\theta \leftarrow \eta_\theta - \alpha\nabla_{\mu_\theta}\mathcal{L}$
  $g \leftarrow \texttt{forwardProp}(\texttt{precToUnc}, \Sigma_\theta, \nabla_{S_\theta}\mathcal{L})$
  $L_\theta \leftarrow L_\theta - \alpha g$

Figure 16: Psuedocode for optimizing the parameters η_θ, L_θ of $q(\theta)$ via natural gradient descent. η_θ is the first natural parameter of $q(\theta)$ (equal to the precision matrix times the mean) and L_θ is an unconstrained reparameterization of the precision matrix. The initial divide-by-100 ensures that the initial posterior $q(\theta)$ is centered around the mean of $p(\theta)$ but has significantly smaller variance to prevent the sampling of extreme values of θ during the first iteration. Optimization guides the posterior away from this initialization.

of the same shape as the input. `forwardProp` then computes a Jacobian-vector product, pushing the perturbations through the function at the input.

- `requiresGrad` represents the Pytorch function which creates a leaf node in a computation graph. It is included in the following algorithms to make them easier to follow. Later in each algorithm, derivatives will be taken with respect to the tensors outputted by `requiresGrad`. This means that all computation which uses these tensors happen *inside* automatic differentiation, and all computation before the `requiresGrad` call happens outside of it (and therefore does not require the memory overhead of automatic differentiation). Similarly, `detach` matches Pytorch's implementation, removing a tensor from the computation graph and treating it like a fixed value. It is necessary to detach θ before optimizing $q(z)$, to avoid storing the entire optimization history for η_θ, L_θ's gradient update step.

Fig. 16 details the outer optimization of the global latent variables θ. It includes as an inner loop the optimization of the state variables x_t, further specified in Fig. 17. Within Alg. 2, some of the arguments to `inferQX` (as specified in Alg. 3) are dropped for brevity. The `calcNELBO` function in Alg. 2 is not specified separately; it is a single iteration of Alg. 3, returning the loss $\mathcal{L}$ instead of using it to update η_x, L_x.

B.1 NATURAL GRADIENT COMPUTATION

Both algorithms make use of the natural gradient technique of Bendekgey et al. [2023]. Prior work [Khan and Nielsen, 2018] has shown that a natural gradient with respect to a distribution's natural parameters is equal to a regular gradient with respect to the distributions moment parameters. For a normal distribution with mean μ and covariance matrix Σ, the moment parameters are

$$E[x] = \mu \qquad \text{and} \qquad E[xx^T] = \Sigma + \mu\mu^T. \tag{25}$$

And the natural parameters are

$$\eta = \Sigma^{-1}\mu \qquad \text{and} \qquad \Lambda = \Sigma^{-1} \tag{26}$$

Therefore, letting $\tilde{\nabla}$ denote the natural gradient, it follows that

$$\tilde{\nabla}_\eta\mathcal{L} = \nabla_\mu\mathcal{L}. \tag{27}$$

Algorithm 3
Inferring $q(x)$

Require: Observation data y, global parameters θ, step size α, number of optimization steps T, number of samples for estimating ELBO M, transformation function $f(\tilde{x})$ that maps unconstrained samples to samples in the support of $p(x)$ (and returns the log absolute determinant of the Jacobian of the transform at $\tilde{x}$).

$\eta_x, L_x \leftarrow \texttt{initializeQX}(\theta)$ `// junction tree or attention initialization`
for $t \leftarrow 1$ to T **do**
  $\Lambda_x \leftarrow \texttt{uncToPrec}(L_x)$
  $\mu_x, S_x, H \leftarrow \texttt{requiresGrad}(\texttt{expectationsViaKalmanSmoother}(\eta_x, \Lambda_x))$
  $\Sigma_x \leftarrow S_x - \mu_x \mu_x^T$
  for $m \leftarrow 1$ to M **do**
    $\tilde{x}_m \leftarrow \texttt{sample}(\mu_\theta, \Sigma_\theta)$
    $x_m, \texttt{logdetJ} \leftarrow f(\tilde{x}_m)$
    $\ell_m \leftarrow -\texttt{logpx}(x_m) - \texttt{logpy}(y, x_m) - H - \texttt{logdetJ}$
  $\mathcal{L} \leftarrow \frac{1}{M} \sum_{m=1}^{M} \ell_m$
  $\eta_x \leftarrow \eta_x - \alpha \nabla_{\mu_x} \mathcal{L}$
  $g \leftarrow \texttt{forwardProp}(\texttt{precToUnc}, \Sigma_x, \alpha \nabla_{S_x} \mathcal{L})$
  $L_x \leftarrow L_x - \alpha g$
return η_x, L_x

Figure 17: Psuedocode for optimizing the parameters η_x, L_x of $q(x)$ via natural gradient descent. η_x is the first natural parameter of $q(x)$ (equal to the precision matrix times the mean) and L_x is an unconstrained reparameterization of the precision matrix.

It also follows that

$$\tilde{\nabla}_\Lambda \mathcal{L} = \nabla_S \mathcal{L}, \tag{28}$$

where $S = E[xx^T]$ as specified above.

Unfortunately, if we take a natural gradient step with respect to Λ, we might end up with a new matrix that is no longer positive definite. Only under strict prior-posterior conjugacy assumptions can we assume this won't happen, and our SDE priors are not conjugate to our Gaussian approximate posteriors. The solution is derived in Bendekgey et al. [2023], which shows that if we want to optimize a parameter of unconstrained values L, and we have

$$\Lambda = T(L) \tag{29}$$

for some transformation function T, then

$$\tilde{\nabla}_L \mathcal{L} = \frac{\partial T^{-1}(\Lambda)}{\partial \Lambda} \nabla_S \mathcal{L}. \tag{30}$$

Computing this natural gradient requires us to take the regular gradient with respect to the moment parameter S, then compute a Jacobian-vector product by pushing that gradient *forward* through T^{-1}, the inverse transformation.

An added benefit of this technique is that computing the moment parameters μ, S from the natural parameters η, Λ happens outside of automatic differentiation and does not require backpropagating through; we only need to push forward the gradient with respect to S. These moment parameters are computed using a variant of the Kalman smoother, which mixes information across time to produce a smooth posterior, without having to perform any iterative-across-time computation inside of automatic differentiation.

Some prior works such as Johnson et al. [2016] compute natural gradients of non-conjugate losses by removing the non-conjugate parts of the loss from the gradient computation, thus obtaining *biased* natural gradients. Our natural gradients however, lack those modifications and are therefore unbiased Monte Carlo estimators of the true natural gradient.

C EXPLOITING THE STRUCTURE OF GAUSS-MARKOV DISTRIBUTIONS

In this section, we provide proofs of central claims of our paper: that Gauss-Markov random variables have block-tridiagonal precision matrices, and that we can compute a precision matrix Λ whose inverse matches observed local covariances efficiently.

C.1 GAUSS-MARKOV DISTRIBUTIONS HAVE SPARSE PRECISION MATRICES

We begin with the general theorem for all (not necessarily Gaussian) random variables.

Theorem 1. *Suppose x is a vector-valued random variable, and let x_i, x_j be scalar components of x. If $x_i \perp x_j | x_{-ij}$, then the i, j entry of $\text{Cov}(x)^{-1}$ is 0.*

Proof. Speed and Kiiveri (1986) prove that the i, j entry of $\text{Cov}(x)^{-1}$ is 0 if and only if x_i, x_j are conditionally uncorrelated given all other components of z. Conditional independence implies conditional uncorrelation, completing our proof. □

An extension of this theorem exists for Gaussian variables:

Lemma 1. *Suppose x is a vector-valued random variable **whose distribution is Gaussian**, and let x_i, x_j be scalar components of x. The i, j entry of $\text{Cov}(x)^{-1}$ is 0 **if and only if** $x_i \perp x_j | x_{-ij}$.*

Proof. Same as Thm. 1, noting that for multivariate Gaussians, variables are conditionally independent if and only if they are conditionally uncorrelated. This is a central theorem of Speed and Kiiveri (1986). □

In a Gauss-Markov distribution, we assume there are D-dimensional observations at each of T time steps. With a bit of notational abuse, we denote the D-dimensional subvector of x corresponding to time step t with x_t. For most rows of the precision matrix, there are $3D$ non-zero entries, because a single component of x_t has no Markov assumption separating it from the D dimensions at the previous, current, and next time step. This structure can be seen on the right of Fig. 2 of the main paper, where light blue blocks of size $2D \times 2D$ reflect the dense set of dependencies between the dimensions of x_t and x_{t+1}.

C.2 FAST MATRIX INVERSION VIA JUNCTION TREE REPRESENTATION

Given the set of local covariances $\text{Cov}(x_t, x_{t+1})$, we want to compute a block tri-diagonal precision matrix whose inverse matches those observed covariances.

Theorem 2. *Let G be a graphical model encoding the Markov assumptions of random variable x. Let $\mathcal{C}$ be the maximal cliques of G, and let $\mathcal{S}$ be the set of separator sets, i.e. maximal sub-cliques which are fully contained by multiple $c \in \mathcal{C}$. Let d_s denote the number of maximal cliques that separator set s is a subset of.*

Given the positive definite covariances $P_c, c \in \mathcal{C}$ of all x components in the same clique c,

1. *There is a unique precision matrix Λ whose inverse matches P_c for all $c \in \mathcal{C}$, and*
2. *Λ can be computed via*

$$\Lambda_{ij} = \sum_{c \in \mathcal{C} | ij \in c} [P_c]^{-1}_{ij} - \sum_{s \in \mathcal{S} | ij \in s} (d_s - 1)[P_s]^{-1}_{ij} \tag{31}$$

Proof. Consider a Gaussian random variable z with mean 0 and covariance J, such that the Markov structure of z is encoded by G. We can write its Gaussian density as follows:

$$p(z) \propto \exp\Big\{ -\frac{1}{2} \sum_{ij} z_i z_j [J^{-1}]_{ij} \Big\}. \tag{32}$$

However, we can also rewrite this distribution using the junction tree decomposition (Cowell et al., 1999):

$$p(z) = \frac{\prod_{c \in \mathcal{C}} p_c(z_c)}{\prod_{s \in \mathcal{S}} [p_s(z_s)]^{d_s - 1}}, \tag{33}$$

where $p_c(z_c)$ and $p_s(z_s)$ are the marginal distributions of those subsets of z. The marginals of Gaussian distributions are Gaussian, with covariance matrices obtained by simply taking sub-blocks of the original covariance matrix. Thus,

$$p(z) \propto \exp\Big\{ -\frac{1}{2}\Big(\sum_{c \in \mathcal{C}} \sum_{ij \in c} (z_i z_j [J_c]^{-1}_{ij}) - \sum_{s \in \mathcal{S}} \sum_{ij \in s} (d_s - 1)(z_i z_j [J_s]^{-1}_{ij}) \Big) \Big\}. \tag{34}$$

Note that this expression has no dependence on the entries of J outside maximal Markov cliques $\mathcal{C}$. Thus, if we set $J_c = P_c, J_s = P_s$ we find the resulting Gaussian distribution in Eq. (34) and therefore the full precision matrix Λ to be uniquely defined.

Deriving the algorithm for computing Λ simply requires us to match coefficients between Eq. (32) and Eq. (34). As they are two different ways of writing the same distribution, the coefficients must match. In Eq. (32), we see entries of the entire precision matrix $\Lambda = J^{-1}$. In Eq. (34), we invert smaller blocks corresponding to the cliques and separator sets of G, and subtract them. Setting $J_c = P_c, J_s = P_s$ we get Eq. (31). □

In the context of a Gauss-Markov distribution, the cliques are of size $2D$, and contain the components at neighboring time steps. The separator sets are of size D, and contain components within a single time step (they are contained in the clique corresponding to $(t-1, t)$ and $(t, t+1)$, and thus $t = 1, t = T$ are not separator sets). We therefore have $|\mathcal{C}| = T - 1$, and $|\mathcal{S}| = T - 2$, allowing us to construct Λ with only $O(T)$ matrix inversions, each of size $O(D) \times O(D)$.

For the existence of the precision matrix Λ constructed by Theorem 2 to be guaranteed, the covariances P_c must be strictly positive definite. Because the graphical model underlying our Gauss-Markov distribution is decomposable, this happens with probability one if the number of prior samples $M > 2D$, as established by Lauritzen [1996, Proposition 5.9].

D NETWORK ARCHITECTURE FOR COVARIANCE PREDICTION

The algorithm for producing a posterior precision matrix is specific in Alg. 18. The only changes from the mean-prediction network are that (i) the mean prediction is taken as input, instead of the (possibly sparse) observations, (ii) the predicted covariance is constructed by a weighted empirical covariance instead of a weighted mean, and (iii) the precision matrix is constructed from a predicted covariance matrix via the block tridiagonal inverse algorithm detailed in Sec. C.

Algorithm 4
Attention-Based Encoder Architecture, Precision Matrix

Require: M prior samples $x \sim p(x|\theta) : M \times T \times D$ and one prediction mean $\mu^* : T \times D$.
$\ell \leftarrow 0, \qquad \tilde{\mu} \leftarrow \texttt{None}, \qquad h^{(1)}_{\tilde{\mu}}(\tilde{\mu}) \leftarrow 0$
for $i \leftarrow 1$ to n **do**
  $\tilde{q} \leftarrow W^{(i)}_Q \left(h^{(i)}_{\tilde{\mu}}(\tilde{\mu}) + h^{(i)}_{\mu^*}(\mu^*) \right)$
  for $m \leftarrow 1$ to M **do**
    $\tilde{k}_m \leftarrow W^{(i)}_K \, h^{(i)}_x(x_m)$
    $\tilde{s}_m \leftarrow \left(\sum_{t=1}^{T} \tilde{q}^T_t \tilde{k}_{mt} \right) / \sqrt{T d_{\text{model}}}$
  $\tilde{\ell} \leftarrow \tilde{\ell} + \tilde{s}$
  $\tilde{w} \leftarrow \texttt{softmax}(\tilde{\ell})$
  $\tilde{\mu} \leftarrow \sum_m \tilde{w}_m x_m$
$P \leftarrow \sum_m \tilde{w}_m (x_m x^T_m) - \tilde{\mu}\tilde{\mu}^T$
return $\texttt{JunctionTreeInverse}(P)$

Figure 18: Specification for estimating the posterior precision given samples from the generative prior and the output of the mean-prediction network, performed in parallel for a batch of B sequences μ^*. All networks share architecture details with the mean-prediction network. Note that not all T^2D^2 dense entries of P are computed; only the $O(TD^2)$ entries requires to apply the junction tree decomposition of Sec. C.

E KALMAN SMOOTHER AND ENTROPY COMPUTATION

The following algorithm details the Kalman smoother used to compute the moment parameters of a Gauss-Markov distribution given the natural parameters. It is adapted from Bendekgey et al. [2023], which requires a redundant parameterized obtained from a Gauss-Markov distribution defined by a linear dynamical system. The entropy of the Gauss-Markov distribution, which depends only on the precision matrix Λ, is computed as part of this inner loop.

Because the entropy is computed outside of automatic differentiation, its contribution to the gradient is not automatically accounted for. Fortunately, the gradient of the entropy with respect to the moment parameters is the natural parameters:

$$\nabla_\mu H = \eta \qquad \text{and} \qquad \nabla_S H = \Lambda \tag{35}$$

Thus we can modify the gradients obtained in Als. 2-3 to correctly account for the entropy by simply adding the natural parameters.

Input: Natural parameters η (vector) and Λ (positive definite precision matrix). Let η_t be the length-D subvector of η corresponding to timestep t, and $\Lambda_{i,j}$ be the $D \times D$ subblock of the precision matrix corresponding to timesteps i and j (note that $\Lambda_{ij} \neq 0$ only when $|i - j| \leq 1$).
Initialize:

$$F_{1|1} = \Lambda_{1,1} \tag{36}$$
$$f_{1|1} = \eta_1 \tag{37}$$
$$H = 0 \tag{38}$$

Filter: for $t = 1, \ldots, T-1$

$$H = H - \frac{1}{2} \log |F_{t|t}| \tag{39}$$
$$F_{t+1|t+1} = \Lambda_{t+1,t+1} - \Lambda_{t+1,t} \cdot F^{-1}_{t|t} \cdot \Lambda^T_{t+1,t} \tag{40}$$
$$f_{t+1|t+1} = \eta_{t+1} - \Lambda_{t+1,t} \cdot F^{-1}_{t|t} \cdot f_{t|t} \tag{41}$$

Finalize:

$$H = H - \frac{1}{2}\log|F_{T|T}| + \frac{DT}{2}(1 + \log 2\pi) \tag{42}$$

$$\Sigma_{T,T} = F_{T|T}^{-1} \tag{43}$$

$$\mu_T = \Sigma_{T,T} \cdot f_{T|T} \tag{44}$$

$$S_{T,T} = \Sigma_{T,T} + \mu_T \mu_T^T \tag{45}$$

Smoother: for $t = T-1, \ldots, 1$

$$C_{t+1} = F_{t+1|T} - F_{t+1|t+1} + \Lambda_{t+1} \tag{46}$$

$$F_{t|T} = F_{t|t} - \Lambda_{t+1,t}^T \cdot C_{t+1}^{-1} \cdot \Lambda_{t+1,t} \tag{47}$$

$$f_{t|T} = f_{t|t} - \Lambda_{t+1,t}^T \cdot C_{t+1}^{-1}(f_{t+1|T} - f_{t+1|t} + \eta_{t+1}) \tag{48}$$

$$\Sigma_{t,t} = F_{t|T}^{-1} \tag{49}$$

$$\Sigma_{t,t+1} = -\Sigma_{t,t} \cdot \Lambda_{t+1,t}^T \cdot C_{t+1}^{-1} \tag{50}$$

$$\mu_t = \Sigma_{t,t} \cdot f_{t|T} \tag{51}$$

$$S_{t,t} = \Sigma_{t,t} + \mu_t \mu_t^T \tag{52}$$

$$S_{t+1,t} = \Sigma_{t,t+1}^T + \mu_{t+1} \mu_t^T \tag{53}$$

Return: moment parameters μ and S, and entropy H of $q(z)$.