# A framework for variational inference and data assimilation of soil biogeochemical models using normalizing flows

**H. W. Xie[1,†], D. Sujono[2,†], T. Ryder[3], E. B. Sudderth[2], S. D. Allison[4,5]**

[1]Center for Complex Biological Systems, University of California, Irvine, Irvine, CA, United States of America
[2]Department of Computer Science, University of California, Irvine, Irvine, CA, United States of America
[3]School of Mathematics Statistics and Physics, Newcastle University, Newcastle, United Kingdom
[4]Department of Ecology and Evolutionary Biology, University of California, Irvine, Irvine, CA, United States of America
[5]Department of Earth System Science, University of California, Irvine, Irvine, CA, United States of America
[†]Authors contributed equally to this work.

**Key Points:**

- Soil biogeochemical models (SBMs) represent the dynamics of soil systems and predict their responses to climate change.
- We fit stochastic SBMs to synthetic data on soil carbon pools and respiration to assess model accuracy and estimate parameter values.
- We develop a variational inference method based on normalizing flows for stochastic SBM data assimilation and parameter estimation.

Corresponding author: Steven D. Allison, `allisons@uci.edu`

## Abstract

Soil biogeochemical models (SBMs) represent soil variables and their responses to global change. Data assimilation approaches help determine whether SBMs accurately represent soil processes consistent with soil pool and flux measurements. Bayesian inference is commonly used in data assimilation procedures that estimate posterior parameter distributions with Markov chain Monte Carlo (MCMC) methods. The ability to account for data and parameter uncertainty is a strength of MCMC inference, but the computational inefficiency of MCMC methods remains a barrier to their wider application, especially with large datasets.

Given the limitations of MCMC approaches, we developed an alternative variational inference framework that uses a method called *normalizing flows* from the field of machine learning. Normalizing flows rely on deep learning to map probability distributions and approximate SBMs that have been discretized into state space models. As a test of our method, we fit approximated SBMs to synthetic data sourced from known data-generating processes to identify discrepancies between the inference results and true parameter values. Our approach compares favorably with established MCMC methods and could be a viable alternative for SBM data assimilation that reduces computational time and resource needs. However, our method has some limitations, including challenges assimilating data with irregular measurement intervals, underestimation of posterior parameter uncertainty, and limited goodness-of-fit metrics for comparison to MCMC inference methods. Many of these limitations could be overcome with additional algorithm development based on the approaches we report here.



## Plain Language Summary

Mathematical models are important tools for predicting how global climate change might affect the storage of soil carbon, a key question in Earth system science. However, current models vary dramatically in their predictions, and scientists need better computer techniques to assess how well the models perform. To address this need, we developed a new technique that uses generative machine learning to optimize and compare soil models with soil carbon data. Our method is flexible and efficient, and it meets or exceeds the performance standards of established techniques for model-data comparison. Although our method looks promising, it needs further improvement to handle real-world data and more sophisticated mathematical models. Our study lays the groundwork for overcoming these limitations so that our method can be applied more widely.


## 1 Introduction

Soil biogeochemical models (SBMs) are differential equation systems that represent dynamics of carbon and nutrient transfers between soil pools, including soil organic (SOC), dissolved organic (DOC), and microbial biomass carbon (MBC). The state variables of SBMs are typically densities or masses of elements in those pools (Manzoni & Porporato, 2009), and fluxes of $CO_2$ can be estimated from those state values and microbial parameters (Allison et al., 2010). As soil microbial communities evolve and shift under the selection pressures of terrestrial warming, SBMs have become an important tool for biogeochemists to quantify changes in soil system activity and predict future soil carbon balance (Sulman et al., 2018; Saifuddin et al., 2021).

To have confidence in SBM predictions, researchers need statistically sound approaches for model validation (Luo et al., 2016; Xie et al., 2020; Bradford et al., 2021; Georgiou et al., 2021; Raczka et al., 2021). The simplest validation technique is visual comparison of empirical observations with model outputs corresponding to manually adjusted parameters (Sulman et al., 2014; Wieder et al., 2015). Correlation tests and root-mean-square errors have also been used for quantitative model evaluation (Todd-Brown et al.,

2013, 2014; Wieder et al., 2014), but these methods do not provide estimates of uncertainty in SBM parameters, $\theta$ (Vehtari et al., 2017; Gelman et al., 2014, 2019)

More advanced data assimilation approaches can estimate parameter uncertainty even for complex models and datasets. For example, Bayesian Markov chain Monte Carlo (MCMC) inference methods (Hararuk et al., 2014; Hararuk & Luo, 2014; J. Li et al., 2019; Xie et al., 2020; Saifuddin et al., 2021; S. Wang et al., 2022) approximate posterior parameter distributions by sampling from ergodic Markov chains. However, the the Gibbs (Geman & Geman, 1987) and No-U-Turn samplers (NUTS) (Hoffman & Gelman, 2014) commonly used with MCMC methods in statistical packages like JAGS (Plummer, 2003), Stan (Carpenter et al., 2017), and PyMC (Salvatier et al., 2016), can be computationally demanding, especially for more complex models (Kucukelbir et al., 2017).

The Kalman filter and its ensemble derivatives including EnKF (Evensen, 1994) are also useful tools for model-data assimilation across fields ranging from weather forecasting to physics and hydrology. The traditional Kalman filter assumes linear model dynamics and Gaussian state distributions for a single, non-ensemble model (Kalman, 1960). Kalman filters and EnKF both operate by sampling from a model state distribution, propagating each sample forward in time, and updating based on observations to refine the state estimates (Katzfuss et al., 2016). EnKF is more flexible because it assembles a state estimate from ensemble constituents with different initial conditions, boundary conditions, forcings, or other model parameters.

EnKF has been applied to data assimilation of nonlinear systems and Earth system data (Keller et al., 2018) with the aim of model parameter estimation, uncertainty quantification, and state predictions. The method seeks to accommodate model stochasticity and non-Gaussian behavior in dynamics and noise, where traditional Kalman filter frameworks fail, through ensemble propagation of Gaussian-driven noise approximations. Importantly though, EnKF may struggle to approximate strongly non-Gaussian noise patterns in stochastic dynamical systems. Standard EnKF implementations do not exactly and analytically represent nonlinear noise parameters governing randomness in equation-based formulations of stochastic dynamical systems that include state space models (SSMs) and stochastic differential equations (SDEs) (Shen & Tang, 2015; Katzfuss et al., 2016; de Bézenac et al., 2020; X. Wang et al., 2025). Consequently, for situations where explicit analytical treatment of non-Gaussian noise is desired during data assimilation, alternative methods may be needed.

Leveraging recent advances in artificial intelligence and deep learning, we investigated variational inference (VI) as an alternative and computationally scalable approach to data assimilation that flexibly admits explicit specification of Gaussian and non-Gaussian noise in dynamical SBMs (Ryder et al., 2021). VI is an approach that frames Bayesian inference as an optimization problem. It optimizes a pre-specified distribution family – which could range from a simple Gaussian to a more complex form parameterized by a deep neural network – to approximate a true posterior resulting from updating a prior with the probability of observations under an observation model (or the likelihood) as closely as possible (Blei et al., 2017). Our VI approach builds on previous work that tested inference approaches for SDEs (Golightly & Wilkinson, 2006; Whitaker et al., 2017; Ryder et al., 2018, 2021).

In contrast to ordinary differential equation (ODE) models, which are completely deterministic, SDE models represent randomness with stochastic *diffusion* parameters (or diffusion terms/coefficients) (Golightly & Wilkinson, 2010; Fuchs, 2013; Hoffman et al., 2013; T. Chen et al., 2014). SDEs provide the flexibility to accommodate noisy observations and underlying stochastic processes without compromising the stability of optimization algorithms used during inference (Whitaker, 2016; Särkkä & Solin, 2019; Wiqvist et al., 2021). Although SDE models are more computationally intensive than their ODE counterparts, advances in graphical processing unit (GPU) architecture and parallel com-

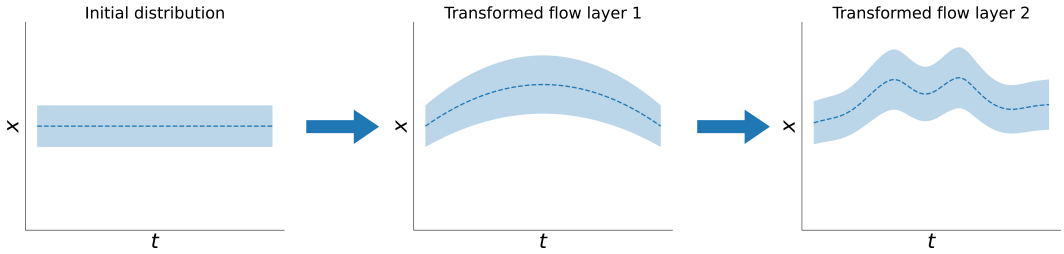


Figure 1: We use normalizing flows to approximate soil biogeochemical model solution trajectories $x$ over time $t$. The flow operates in a generative direction, mapping a simpler initial base distribution to a more complex one representing model output.

puting are making SDE implementation more tractable (Januszewski & Kostur, 2010). SDE models also have advantages in representing the stochasticity inherent to many biophysical processes (Golightly & Wilkinson, 2005, 2011; Abs et al., 2020; Browning et al., 2020).

For our VI approach, we applied a class of methods called *normalizing flows* used in machine learning to generate probability distributions. Flows include one or more layers of random variable mappings that transform an initial base probability distribution into a new distribution (Papamakarios et al., 2021). When transforming a simpler probability density into a more complex one (Figure 1), the flow operates in the generative direction (Kobyzev et al., 2020). We use generative flow approximation to build a probabilistic state space model for data assimilation and inference. SSMs represent the distributions of noisy observations $y$ layered over equations (SDEs in our case) for the *latent state* variable paths $x$ (Särkkä & Solin, 2019). The latent state is an unobserved set of variables that represent the underlying dynamics of the SBM. The SDEs of the SBM serve as known data-generating processes whose true $\theta$ values can be compared with the posteriors inferred from the flow-based VI method.

Unlike VI's optimization framing, established MCMC-based Bayesian inference approaches, including NUTS, treat posterior estimation as a sampling problem in which posteriors are estimated through the nonparametric accumulation of posterior samples. For these methods, posterior distributions are not constrained to pre-specified parametric or semiparametric distribution families, but this posterior freedom can increase computational expense.

Using a proof-of-concept approach, our aim was to establish a conceptual foundation and initial implementation of VI for SBMs. Although we will ultimately need to validate SBMs and the VI approach with empirical data, at this proof-of-concept stage, we used synthetic data to avoid temporal gaps and unknown sources of error inherent in real-world data. We conditioned our VI on these synthetic observations to estimate approximate posterior densities $q(\theta)$ that were compared to prior densities $p(\theta)$. Using a set of simulation experiments, we addressed the following questions: 1) How do results from our normalizing flow VI approach compare to those from established inference approaches? 2) How does the availability of measurements affect posterior parameter identifiability and distributions obtained from VI? 3) How does SSM-flow perform with a moderately more complex model featuring a nonlinear noise structure that may be more representative of stochastic real-word processes?

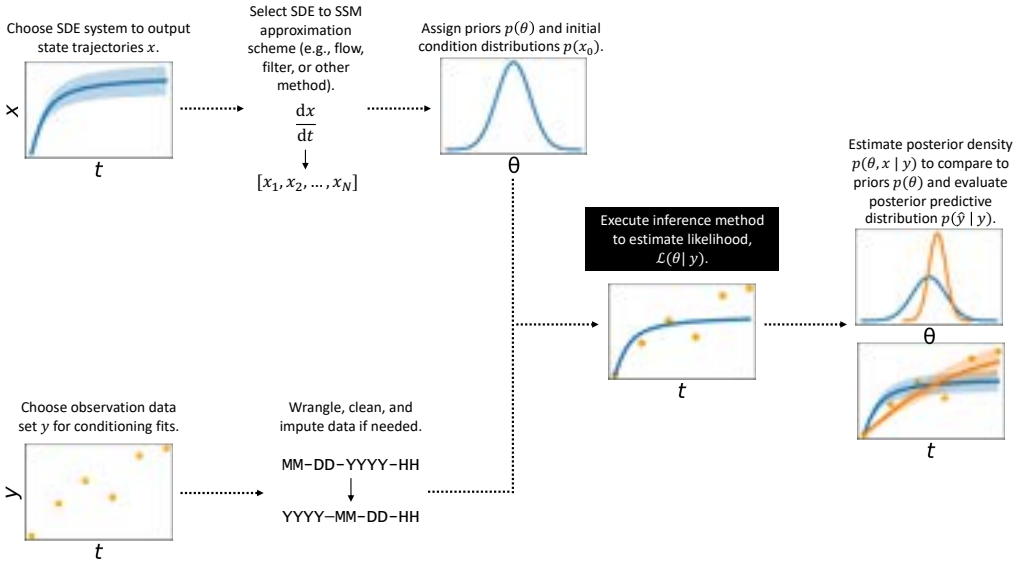


Figure 2: A workflow diagram summarizing the steps involved in our study's stochastic variational Bayesian framework. Our workflow conducts inference and data assimilation on stochastic differential equation (SDE) soil biogeochemical models (SBMs) with their approximation into state space models (SSMs). Our modular workflow is designed to serve as a basis for building future soil biogeochemical model inferences, as the inference method used can be modified or substituted. Our inference method of choice was stochastic gradient descent mean-field variational inference. Within the nodes of the diagram, blue lines and shading correspond to prior means and distributions, while orange lines and shading correspond to posterior means and distributions. Orange dots represent observations upon which the inference is conditioned.

# 2 Materials and Methods

## 2.1 Soil biogeochemical model formulation

We formulated an SDE version of the linear deterministic "conventional" (CON) soil biogeochemical model (Allison et al., 2010; J. Li et al., 2014). CON is a simple, well-studied SBM in which dead plant material enters SOC and DOC pools. Microbial biomass takes up DOC and releases $CO_2$. Transfer coefficients determine the movement of carbon between pools, all of which decay according to linear, first-order dynamics. As with CON, our stochastic version of the model, SCON, has three state variables representing SOC, DOC, and MBC densities. We used decay parameters consistent with the relatively fast dynamics of organic matter at the soil surface rather than the slower dynamics typical of deep soil (J. Li et al., 2014; Xie et al., 2020). This parameterization reduces computational intensity by permitting the use of shorter data sets $y$ that still contain the dynamical richness needed for robust posterior estimation by our inference algorithm.

We parameterized two SCON model variants, "constant diffusion" (SCON-C) and "state-scaling diffusion" (SCON-SS). These models include diffusion coefficients that govern the magnitude of stochastic noise, or variance, in an SDE model (not to be confused with the chemical definition of diffusion related to molecular movement). In SCON-C, diffusion was set to be independent of time and states, while in SCON-SS, diffusion depended on and scaled with state values. Details of these parameterizations are given in section 5.1. In the real world, constant diffusion might correspond to a physical data-generating process whose variance is not influenced by the magnitude of the process variable. State-scaling diffusion would correspond to a data-generating process whose variance is related to its process variable magnitude.

## 2.2 Synthetic data generation

Synthetic data $y$ were simulated from our data-generating SDE model, SCON, parameterized in time units of hours. Observations of soil state variables and $CO_2$ were collected every 5 hours by default for up to 5000 hours in total time $T$. Our approach to state space approximation of SDE output requires regular time series data (Kalman, 1960) such that all observations coincide with discrete time steps of the SSM. Different SDE approximation methods would be needed for irregular time discretization in which time steps between latent states are not equivalent. Our data generation, prior, and posterior distributions were assumed to be logit normal with truncation between lower and upper support bounds (section 5.2). To simulate fast decay at the soil surface, we used relatively high decay coefficient $k_{i,\,\mathrm{ref}}$ means compared to previous literature values (Allison et al., 2010; J. Li et al., 2014; Xie et al., 2020). All $\theta$ for data generation, including the hyperparameters of the prior distribution $p(\theta)$, are given in Table S1 together with their biogeochemical interpretations.

## 2.3 Variational inference

The key steps our SBM data assimilation workflow are outlined in Figure 2 and Algorithm 1. We selected mean-field stochastic VI as our inference method for its mathematical simplicity and flexibility. Mean-field inference assumes that model parameters are independently distributed. VI frames Bayesian inference as an optimization goal of finding an approximate posterior from a chosen posterior family that is the closest estimate to the true posterior. The optimization objective is a metric called the evidence lower bound (ELBO). In the case of mean-field VI, the posterior family consists of independently distributed $\theta$ densities. To simplify computation, we chose to use independent Gaussians in our synthetic data-generating process and for our approximate mean-field posterior family. After updating SBM $\theta$ and hidden neural network parameters using gradient descent, Algorithm 1 samples $\theta$ values at each training iteration and com-

**Algorithm 1:** Synopsis of operations occurring in each iteration of our soil biogeochemical SSM VI framework

**Data:** Time series matrix $y$ of soil pool state and other observations, including $CO_2$ respiration measurements

**Result:** Posterior approximation $q(\theta, x; \phi_{(\theta,x)})$ following stochastic gradient optimization based on the evidence lower bound (ELBO) $\mathcal{L}[\phi_{(\theta,x)}]$, where $\theta \leftarrow$ model parameters; $x \leftarrow$ latent state variables; $\phi \leftarrow$ posterior parameters;

Define $q(\theta; \phi_\theta)$ and $q(x|\theta; \phi_x)$;
Initialize $(\phi_\theta, \phi_x)$;
$n \leftarrow$ total desired training iterations;
**for** $i \leftarrow 1$ **to** $n$ **do**
  **for** $s \leftarrow 1$ **to** $\mathcal{S}$ **do**
    $s$ denotes an individual sample from a distribution;
    Draw $\theta^{(s)} \sim q(\theta; \phi_\theta)$;
    Draw element $z^{(s)}$ from a standard normal distribution and transform to $x^{(s)} \sim q(x|\theta; \phi_x)$;
  **end**
  Compute $\mathcal{L}[\phi_{(\theta,x)}]$ (or $-\mathcal{L}[\phi_{(\theta,x)}]$ for gradient descent) as per equation (A73);
  Compute the gradient $\nabla\widehat{\mathcal{L}[\phi_{(\theta,x)}]}$ with automatic differentiation;
  Reinitialize variational parameters $\phi_{(\theta,x)}$ based on the gradient;
**end**

putes the likelihood of the resulting model output conditioned on $y$. This algorithm searches for parameters that maximize model likelihood relative to the given $y$.

We used normalizing flows to approximate SCON-C and SCON-SS as SSMs with discrete time steps (section 5.3) instead of sampling observations from forward-solved SDE trajectories to formulate SSMs. Fundamentally, flows are probability distributions created by deep neural network layers that transform probability densities from simple to complex or vice versa. These invertible, reversible transformations are called *bijections*, and they "flow" from one layer to the next. In our case, the flow operates in a generative direction with simple inputs transformed into more complex distributional outputs. Invertible bijections mathematically allow for complex and expressive flow outputs without sacrificing feasibility in computing likelihoods of flow samples conditioned on observations (Ryder et al., 2018; Kingma et al., 2016; Papamakarios et al., 2017). A flow learns to serve as an "adjoint" or "backwards-solved" proxy for the SDE without the need for computationally expensive forward-solving. At each inference iteration, our SSM approximation simultaneously samples multiple $x$ (also known as *latent variables*, *states*, or *paths* in the machine learning literature) in one simultaneous draw from the state space distribution learned by the flow up to that point of computation. Information from SCON (equation A3) is incorporated into flow training and optimization through the determination of latent state adjoint transition likelihoods (the probability of jumps between sampled latent states) based on current $\theta$ (R. T. Q. Chen et al., 2018; Ryder et al., 2018).

These flow-drawn state space approximations served as our foundation for VI optimization (section 5.4). Based on the ELBO as shown in equation A31, VI minimizes the discrepancy between an approximate parametric posterior and the true posterior distribution (Salimans et al., 2015) using gradient descent for optimization. Differences between the priors and posterior densities also provide information on how well the data $y$ identify and constrain posterior parameters. The constituent pieces of the machine learning architecture underlying our SSM-flow method are detailed in section 5.5. Masked 1-

D convolutions constitute the core of the flow and the hyperparameters of those convolution transformations are learned by the algorithm.

Under GPU VRAM memory resource limits, we settled on using 5 layer sets of permutation, affine, and batch renormalization layers to comprise our neural moving average flow. This number offered qualitatively smoother and superior fits over flow architectures with lower layer set counts. For inferences conditioned on $y$ of duration $T = 5000$ with $\Delta t = 1.0$ with 5 layers, maximum training batch size $\mathcal{S}$ at 16 GB of VRAM was 31, so we set $\mathcal{S} = 31$. Less memory-limited inferences conditioned on $y$ of lower $T$ afforded larger batch sizes.

We used AdaMax (Kingma & Ba, 2015) for gradient descent optimization. This method requires initial specification of the *learning rate*, a hyperparameter that scales the objective gradient and regulates the extent to which neural network weights can be updated with each training iteration. We began optimization with a *training warmup* phase that used a low learning rate of 1e-6. Consistent with previous studies (Goyal et al., 2017), we found that starting with 5000 to 20,000 warmup iterations allowed us to use higher overall learning rates, experience more stable ELBO trajectories, and converge to lower average ELBO values during training (Figure S1). Following warmup, the learning rate is typically raised and then reduced to promote convergence of the objective function (You et al., 2019). For inference experiments with $T = 5000$, we used a maximum ELBO learning rate of 5e-3 for joint optimization (optimizing both $\theta$ and $x$) and 3.8e-4 for fixed-$\theta$ optimization (only optimizing $x$ with knowledge of true $\theta$), reducing the learning rate by a factor of 0.7 per 10,000 iterations. Joint optimization used 150,000 regular training iterations with 10,000 warmup iterations, whereas fixed-$\theta$ optimization used 240,000 regular training iterations with 20,000 warmup iterations. Joint and fixed-$\theta$ optimization required different iteration levels to maximize neural network stability.

### 2.4 Experimental design for variational inference testing

| Experiment | Description | Figure(s) |
|---|---|---|
| SCON-C SSM-flow | Constant diffusion, state-space model with normalizing flow and 5000-hour duration | 3, 4 |
| SCON-C Kalman smoother | Benchmark of the constant diffusion SDE model state resolved with Kalman smoother and 5000-hour duration | 3, 4 |
| SCON-C SSM-NUTS | Constant diffusion, state-space model with MCMC No-U-Turn sampler and 5000-hour duration | 3, 4 |
| SCON-C SSM-flow no $CO_2$ | Constant diffusion, state-space model with normalizing flow, 5000-hour duration, and no $CO_2$ data assimilation | S5 |
| SCON-C SSM-flow short duration | Constant diffusion, state-space model with normalizing flow and 1000-hour duration | S6 |
| SCON-C SSM-flow high frequency | Constant diffusion, state-space model with normalizing flow and 1000-hourly time points | S6 |
| SCON-SS SSM-flow | State-scaling diffusion, state-space model with normalizing flow and 5000-hour duration | S7, S8, S9 |
| SCON-SS SSM-NUTS | State-scaling diffusion, state space model with MCMC No-U-Turn sampler and 5000-hour duration | S7, S9 |
| SCON-SS SSM-flow reduced parameters | State-scaling diffusion, state-space model with normalizing flow, 5000-hour duration, and some parameters fixed | 5 |

Table 1: Simulation experiments used in developing and testing flow-based variational inference.

To benchmark our flow approximation method before applying it to more complex models like SCON-SS, we visually compared SCON-C SSM-flow $x$ to the true $x$ solution computed by the Kalman smoother algorithm (Table 1, section 5.6). We did not compare the SCON-SS flow output to a Kalman smoother "ground truth" because the Kalman algorithm cannot resolve models with nonlinear diffusion structures. Given a known data-generating process and observation error, a Kalman smoother exactly solves the true mean latent path $x$ of the SDE data-generating process sourcing $y$ (Särkkä, 2013). When an SSM is linear in drift and its diffusion is stationary and additive, as is the SSM approximation of SCON-C, the posterior density $p(x|y)$ can be determined analytically and precisely in closed form with the Kalman smoother algorithm, provided the algorithm is fed the true $\theta$ and observation noise (Kalman, 1960; Rauch et al., 1965).

We also compared our SSM-flow performance to NUTS, an established MCMC method (Hoffman & Gelman, 2014). NUTS is an extension of the original Hamiltonian Monte Carlo sampler described in Duane et al. (1987) that places additional controls over the proposal of MCMC transition steps (Hoffman & Gelman, 2014). A description of the intuition motivating the NUTS and Hamiltonian Monte Carlo algorithms can be found in Betancourt (2017).

To assess the impact of observation availability and frequency on flow-VI performance with SCON-C, we varied the length and information content of $y$ (Table 1). In one data scenario, we included only soil state variables and omitted the $CO_2$ time series from $y$. Preliminary inference tests using only $CO_2$ for data assimilation were unsuccessful at constraining $\theta$, so we included state observations in all subsequent analyses. The second scenario reduced the duration of $y$ from 5000 to 1000 hours. The third scenario also reduced the duration to 1000 hours but increased the observation frequency to hourly instead of 5-hour intervals. We evaluated the impact of these $y$ scenarios on SCON-C posterior parameter distributions and identifiability relative to the baseline SSM-flow scenario with $CO_2$ included and 5000 hours of observations at 5-hour intervals.

After testing flow-VI performance with SCON-C, we tested the method with SCON-SS, a more complex model with nonlinear diffusion (Table 1). We attempted to compare the flow method with NUTS, analogous to our benchmarking analysis of SCON-C. However, we experienced convergence issues and long computational times (over two weeks) with the NUTS method, so the resulting posteriors may not be comparable to those from the flow method for SCON-SS. In an effort to distinguish parameter identifiabilty issues from problems with our flow-VI method, we established a reduced SCON-SS model with all $\theta$ fixed in value except for the $k_{i\in\{S,D,M\}}$ linear decay and state-scaling $\beta$ diffusion parameters. Mirroring above procedures, a synthetic $y$ with $T = 5000$ was produced by a reduced SCON-SS data-generating process to condition an SSM-flow optimization. We expected that if our inference algorithm was functioning properly, fixing some parameters would improve identifiability of the remaining free parameters.

### 2.5 Software and hardware

With respect to the computational software and hardware powering the inference operations, our DGP and inference code was developed for a Python 3.9.7 environment distributed by Anaconda (*Anaconda Software Distribution*, 2020) and used the Numpy 1.20.3 (Harris et al., 2020) and PyTorch 1.10.2 (Paszke et al., 2019) software libraries. PyTorch 1.10.2 was compiled with the Nvidia CUDA 10.2 toolkit. The inferences were run on one Nvidia Tesla V100 GPU at a time updated to CUDA version 11.4.0 with a maximum of 16 GB of video random access memory (VRAM) and two Intel Xeon Gold 6148 CPU cores clocked at 2.40 GHz on the University of California, Irvine HPC3 cluster. For our NUTS benchmark comparisons, we used the NUTS implementation from the hamiltorch Python package (Cobb, 2023).

# 3 Results

## 3.1 SCON-C SSM-flow performance and comparison to benchmarks

Our flow-based VI approach generated SCON-C state trajectories that matched the temporal variation in sampled data, $y$. The flow VI algorithm converged to an ELBO minimum based on stabilization of $-\mathcal{L}[\phi_{(\theta,x)}]$ between $-1600$ and $-1650$ in the latter half of variational training iterations (Figure S2). After ELBO training, the mean of the marginal posterior density of latent states $q(x|\theta;\phi_x)$ was estimated from 250 $x$ samples drawn from the joint variational density. The mean latent SOC, DOC, and MBC paths and state-derived $CO_2$ measurements corresponding to the SCON-C flow fell within the $y$ observation noise throughout the time series (Figure 3a). The latent means matched many of the sharp peaks and valleys in the dynamics of the data, and the flow $CO_2$ mean reproduced the rapid oscillatory behavior of the observed $CO_2$ time series.

Compared to Kalman smoother and SSM-NUTS benchmarks, our SSM-flow method showed satisfactory performance. The flow means for SOC, DOC, and MBC almost always remained within the the 95% Kalman posterior interval throughout the 5000-hour time series ($t = 0$ to 500 shown in Figure 3b). Also, the $CO_2$ mean and distribution closely matched their Kalman counterparts. The extent of the SOC, DOC, and MBC posterior uncertainties were underestimated by the flow (orange band) compared to SSM-NUTS and the Kalman smoother (purple and green bands). This result is consistent with literature showing that the mean-field VI approach underestimates posterior uncertainty due to the assumption of uncorrelated model parameters (Kucukelbir et al., 2017; Sujono et al., 2022; Lambert et al., 2023). Still, the overall alignment of latent state densities with NUTS and the Kalman smoother lends confidence to the SSM-flow method.

The SSM-flow method also performed well based on quantitative comparisons of mean log joint probability, $\log p(\theta, x|y)$ (also known as the mean log joint posterior density). In both VI and NUTS, log joint probability is an equivalent, relative metric of model optimization progress that describes the compatibility of models with prior knowledge and observed data. Log joint probability can only be used for model comparison when the observed data and priors are equivalent. Each method in SSM-flow and SSM-NUTS converges toward higher joint probability as posteriors that better describe the observations are optimized or sampled. With $\theta$ fixed to facilitate comparison with the Kalman smoother as shown in Figure 3b, log joint probability (notated as $\log p(x, y; \theta)$) was 25140 for the SSM-flow and 25380 for SSM-NUTS (Figure S3). We also obtained favorable results when $\theta$ was not fixed, with the log joint probability stabilizing more quickly for SSM-flow than for SSM-NUTS (Figure S4).

Consistent with the fit to synthetic data shown in Figure 3, almost all marginal posterior densities narrowed compared to the SCON-C priors with the information learned from $y$ by the flow algorithm (Figure 4). Moreover, most of the marginal $q(\theta;\phi_\theta)$ means moved closer to the true $\theta$, including the means of $u_M$, $a_{SD}$, and $k_{S,\,\mathrm{ref}}$. The parametric posterior densities of the SSM flow $q(\theta;\phi_\theta)$ also aligned well with the nonparametric posterior densities $p(\theta|y)$ estimated by the SSM-NUTS approach. The posterior density modes of $u_M$ and $k_{D,\,\mathrm{ref}}$ were close to their true values in both the SSM-flow and SSM-NUTS simulations. Posterior modes fell on either side of the truth in the case of $a_{DS}$ and $a_M$. For both approaches, the posterior modes narrowed and shifted in the same direction for $k_{M,\,\mathrm{ref}}$, $Ea_S$, $Ea_D$, and $Ea_M$. For $k_{S,\,\mathrm{ref}}$, the SSM-flow posterior mode consolidated at the true value unlike the SSM-NUTS mode. For $a_{MSC}$, the SSM-flow posterior mode changed little from the prior and remained poorly informed, while the SSM-NUTS mode consolidated away from the true value.

Uncertainties in posterior parameter distributions were generally smaller with SSM-flow compared to SSM-NUTS. This outcome is likely due to the tendency of mean-field VI to underestimate parameter uncertainty, as it assumes parameters are uncorrelated

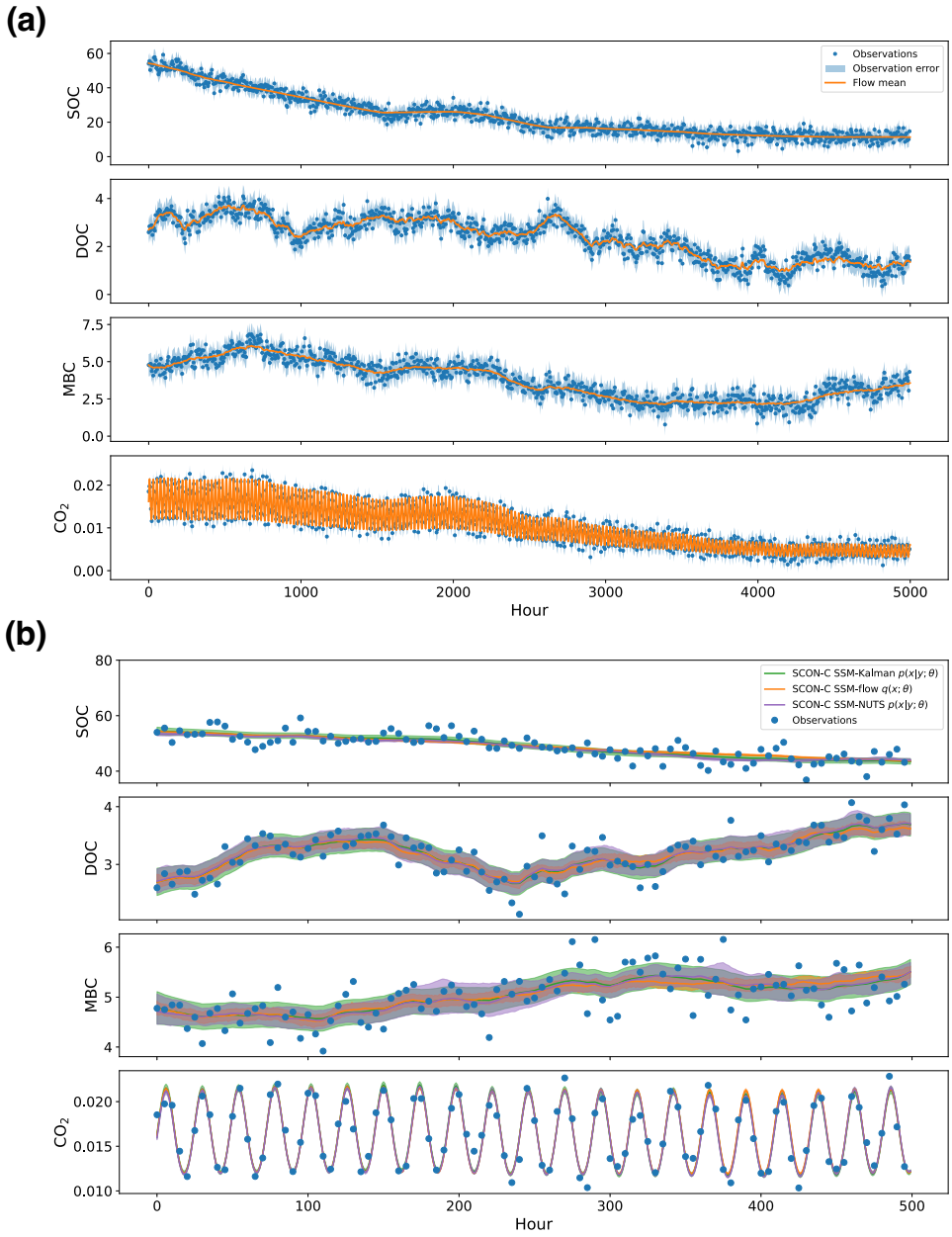


Figure 3: SCON-C SSM state trajectories. The states are in units of mg C $g^{-1}$ soil. In **(a)**, we examine the fit of the means of SSM-flow-approximated state trajectory densities (orange), where hidden neural network parameters and SCON-C $\theta$ were jointly optimized, to synthetic observations $y$ from an SCON-C $T = 5000$ hour data-generating process. The synthetic $y$ is backgrounded by the 95% interval of the observation noise (blue). In **(b)**, we compare the SCON-C SSM-flow-approximated state means and 95% credible interval (orange) against the state means and credible intervals of the SSM-NUTS approximation and the "ground truth" computed by a Kalman smoother (green) with a zoomed-in plot from $t = 0$ to 500. As the smoother was given knowledge of the true SSM $\theta$ for its operation, the SSM-flow and SSM-NUTS $\theta$ were also fixed for the flow such that only the flow's hidden neural network parameters were optimized. This fixing of $\theta$ is notated in the legend with the use of "$;\theta$" indexing. SSM-flow state densities were estimated from 250 $x$ samples drawn at a time from the final optimized joint densities.

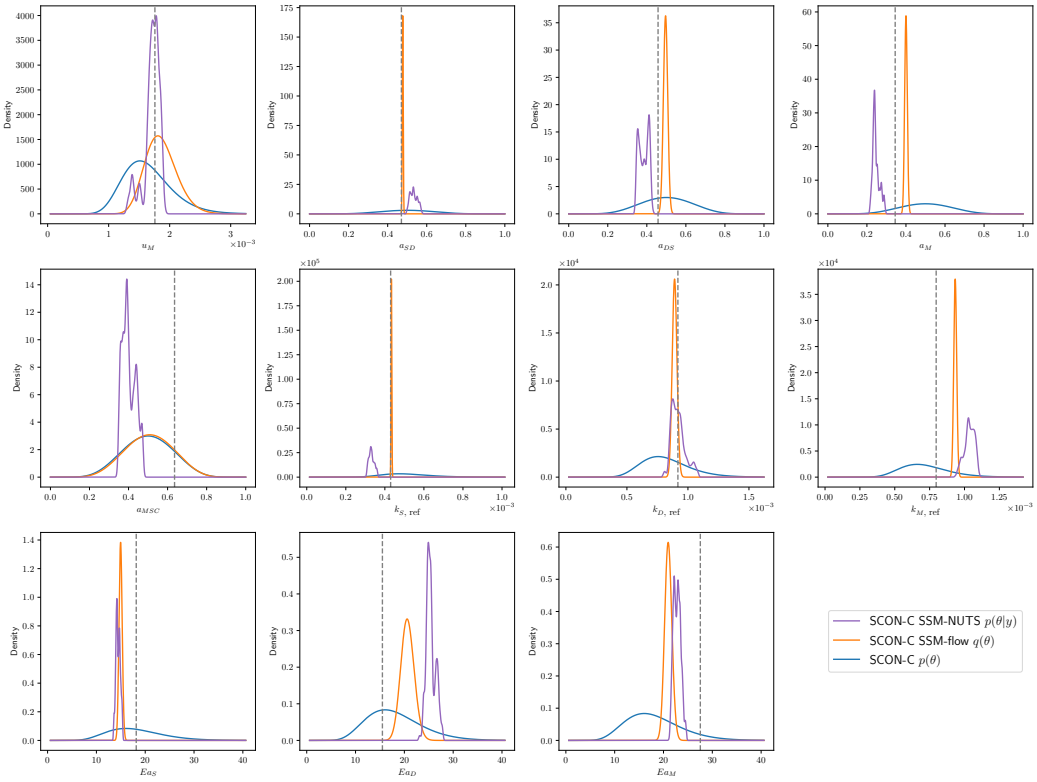


Figure 4: SSM-flow $q(\theta;\phi_\theta)$ marginal posterior densities (orange) compared to the mean-field prior $p(\theta)$ (blue), non-parametric SSM-NUTS densities (purple), and sampled true $\theta$ values used for synthetic data generation (gray). SSM-flow and SSM-NUTS were conditioned on the same $T = 5000$ hour $y$ generated by an SCON-C SDE. The $\beta$ diffusion parameters $c_S$, $c_D$, and $c_M$ were left off this figure to maximize readability and focus on the drift $\theta$ governing the biogeochemical dynamics.

(Sujono et al., 2022). While preliminary analysis of SCON-C NUTS posterior samples demonstrated some posterior correlations, all correlations were below 0.5 in magnitude with most under 0.25, suggesting the approximating assumption of parameter non-correlation was reasonable and plausible a posteriori.

### 3.2 Impact of changing information in $y$ on SCON posteriors

We tested how SCON-C posteriors were affected by changes in the information content of $y$. Without $CO_2$ information, marginal $q(\theta;\phi_\theta)$ posterior densities tended to be wider and less informed, and posterior modes were frequently farther away from the true $\theta$, as illustrated by $a_{SD}$ and $Ea_S$ (Figure S5).

Reducing the amount of information in $y$ by shortening the duration of the time series from 5000 to 1000 hours reduced posterior identifiability (Figure S6). With the shorter time series, $q(\theta;\phi_\theta)$ posteriors became more uncertain for all $\theta$. The modes of some posterior parameters including $a_{SD}$, $k_{S,\,\mathrm{ref}}$, $k_{D,\,\mathrm{ref}}$, $Ea_S$, and $Ea_M$ deviated further from their true values. However, the modes of $\beta$ diffusion coefficients $c_S$, $c_D$ and $c_M$ were smaller and closer to their true values when inferred from the shorter $T = 1000$. With a shorter time series, there is less growth in cumulative approximation error of state space $x$ trajectories, leading to lower diffusion $\theta$ estimates during inference.

If we set $T = 1000$ while increasing observation frequency to hourly (instead of once every 5 hours), parameter identifiability improved overall (Figure S6). With the exception of $k_{S,\,\mathrm{ref}}$, parameter means remained similar or moved closer to their true values relative to the $T = 1000$ case with observations every 5 hours. Some parameter modes, including the diffusion coefficients, were closer to their true values compared to the $T = 5000$ case. This result suggests that more frequent observations can at least partly compensate for shorter time series duration when estimating $\theta$.

### 3.3 SSM-flow application to SCON with state scaling diffusion

Our SSM-flow method also performed well when applied to SCON-SS, a more complex SDE model with state-scaling diffusion (Figure S7). As with the SCON-C SSM-flow, the mean latent state trajectories of the trained SCON-SS SSM-flow followed the state observations (Figure S8). The trajectories matched the peaks and valleys of the state dynamics in $y$, and the flow $CO_2$ mean derived from the sampled states replicated the magnitude of the $CO_2$ oscillations. Compared to the priors $p(\theta)$, the marginal $q(\theta;\phi_\theta)$ densities narrowed with most posteriors moving toward their true values (Figure S9). Although we attempted to compare SSM-flow and SSM-NUTS methods applied to SCON-SS, we experienced issues with NUTS convergence (Figure S7), which reduced our confidence in the NUTS posterior estimates (Figure S9).

Some posterior modes obtained from SCON-SS SSM-flow remained distant from their true values (Figure S9). To rule out issues with the flow algorithm, we fixed a subset of $\theta$ and tested whether identifiability of the remaining $\theta$ improved (Figure 5). As expected, the marginal $k_{S,\,\mathrm{ref}}$ $q(\theta;\phi_\theta)$ posterior density was tightly constrained near the true $k_{S,\,\mathrm{ref}}$ value. The $k_{D,\,\mathrm{ref}}$ and $k_{M,\,\mathrm{ref}}$ posterior densities narrowed substantially, although the means did not align exactly with their true $\theta$, suggesting that there were still some limitations on parameter identifiability.

We found that the $s_{i\in\{S,D,M\}}$ diffusion $\theta$ posteriors consolidated at higher values than the priors (Figure 5). Elevated diffusion $\theta$ posteriors were also observed for the SCON-C approximation (Figure S5). This result does not necessarily indicate a problem with the flow VI method. Rather, the flow neural network approximation algorithm is designed to introduce approximation error that accumulates and affects the diffusion $\theta$ estimates.

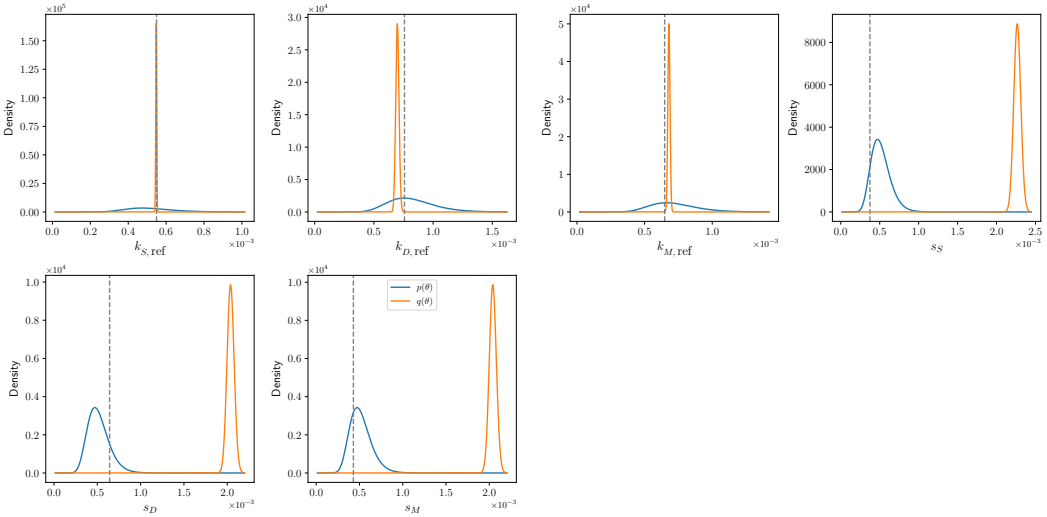


Figure 5: Optimized marginal posterior $q(\theta;\phi_\theta)$ densities (orange) of a reduced SCON-SS model with all but the $k_{i\in\{S,D,M\}}$ decay and state-scaling diffusion $\theta$ fixed compared to mean-field priors $p(\theta)$ (blue). Gray dashed lines show sampled true $\theta$ values used for synthetic data generation.

## 4 Discussion

We developed a stochastic SBM data assimilation and inference framework that is versatile, stable, and computationally tractable, especially when GPU hardware is available. Our approach approximates SBMs as SSMs with state trajectory sampling at reduced computational and temporal cost in comparison to SDEs. Our results suggest that the neural moving average flow could approximate the SCON family of SDE systems as SSMs, fit $y$, constrain posteriors, and recover some true $\theta$ values. We demonstrate that flow approximation and optimization performance compares favorably to a benchmark MCMC procedure in SSM-NUTS conditioned on the same $y$.

We carried out SSM approximation with a class of normalizing flows called neural moving average flows that successively transition random variables from simpler to more complex distributions through invertible transformations. Importantly, normalizing flows preserve likelihood computations conditioned on observed data, which is critical for probabilistic inference. We applied this SSM-flow approach to fit approximated representatives of the SCON family of SBMs to synthetic data. Conditioning with synthetic data allowed us to visualize discrepancies between estimated posterior densities, data-generating densities, and true $\theta$ values used by the data-generating process.

SCON-C state trajectories approximated by SSM-flow tracked state and $CO_2$ observations, aligned with the true latent state distributions determined by a Kalman smoother, and compared favorably against SSM-NUTS-approximated states and $\theta$ posteriors (Figures 4, 5). Following validation of our SCON-C inference against Kalman and SSM-NUTS benchmarks, we extended our approach to SCON-SS (Figures S7, S8, S9), which is more complex than SCON-C due to its nonlinear diffusion terms. A direct comparison between SSM-flow and the NUTS method was not possible due to poor NUTS convergence with SCON-SS, suggesting that the SSM-flow method may be computationally preferable for use with more complex state scaling models.

### 4.1 Constraining $\theta$ posteriors recovered by SSM-flow

Our analysis revealed challenges with SSM parameter identifiability, a common issue for complex biogeochemical models. Nonetheless, we chose to analyze SCON to facilitate comparison to prior work (Allison et al., 2010). Our results suggest that the issues with parameter identifiabilty stem from the SCON model parameterization rather than problems with our VI algorithm. For example, when we simplified the SCON-SS model by fixing a subset of $\theta$, parameter identifiability of decay rates improved, although there were still minor deviations from the true parameter values (Figure 5).

For more complex models, additional measurements, such as radiolabeled C densities, could further constrain and identify $\theta$ posteriors, especially SBM pool transfer coefficients. In our analysis, assimilating $CO_2$ data somewhat improved parameter identifiability (Figure S5), although not enough to identify the marginal posterior of the SCON MBC-to-SOC transfer $\theta$, $a_{MSC}$ (Figures 4, S9). The $a_{MSC}$ parameter appears in two terms of equation (A3) that are each the product of four elements, the $a_M \cdot a_{MSC} \cdot k_M \cdot M$ term in the d$S$ equation denoting C mass transfer from the MBC to SOC soil pool and the $a_M \cdot (1 - a_{MSC}) \cdot k_M \cdot M$ term in the d$D$ equation denoting C transfer from the MBC to DOC soil pool. The posterior densities of the $a_M$ and $k_M$ $\theta$ in those terms appear in equation (A10) and are accordingly better constrained and identified with $CO_2$ measurements in $y$. This is not the case for $a_{MSC}$, which is not present in equation (A10). Constraining $a_{MSC}$ therefore depends on state measurements in $y$, and as only one element in the drift product terms, $a_{MSC}$ can take different values within its $[0, 1]$ support bounds without greatly affecting the results.

### 4.2 Limitations and future directions

Although we demonstrated functional flow VI with synthetic data, a key next step is to modify our approach to accommodate real data, such as multivariate observations from long-term ecological research (LTER) experiments (Melillo et al., 2017; Wood et al., 2019). Assimilating these data sets remains a challenge because of their long duration and irregular measurement intervals. LTER experiments can run 100,000 to 200,000 hours, much larger than the $T = 5000$ hours we tested in our study. Even with GPU hardware, our $T = 5000$ inferences typically required one or two days to converge, but even more limiting than time were GPU memory thresholds preventing adequate variational sample sizes with $T$ much longer than 5000 hours at $x$ simulation discretizations sufficiently fine to mitigate numerical error propagation and loss of of dynamical fidelity. Our experience was that $x$ simulation time span was more important for inference runtime than observation frequency.

One option for conditioning inferences on $y$ corresponding to longer time spans to minimize inference runtime and VRAM consumption is to avoid simulating SSM $x$ for the entire observation time span at each training iteration by applying the *mini-batching* technique. With mini-batching, $y$ and $x$ are partitioned into smaller subsets $y_{\tau i}$ and $x_{\tau i}$ during training, where $\tau$ distinguishes a subset from the entire sequence and $i \in \mathcal{B}$ where $\mathcal{B} \in \mathbb{N}$ is the set of integers counting total subsets in natural order. In each training iteration, a $y_{\tau i}$ is randomly selected for likelihood evaluation such that the SBM only needs to simulate $x_{\tau i}$ sub-samples to calculate the optimization objective. Mini-batching could be integrated into our framework, having been applied in recent flow-related machine learning literature (Papamakarios et al., 2021; Ryder et al., 2021).

Data sets from most biogeochemical studies vary in measurement intervals due in part to constraints on data collection procedures. Whereas soil respiration may be measured hourly, soil carbon pools tend to be observed on monthly to decadal timescales consistent with their slower dynamics (Melillo et al., 2017). Some existing data assimilation tools, such as ensemble Kalman filters and 4D-Var, can handle irregular measurement intervals, but current flow VI and state space models assume regular observations. Fur-

ther development of these algorithms will be needed to accommodate irregularly spaced data. For example, the flow architecture could be modified to accommodate dynamic time steps that effectively skip over periods lacking observations, during which adjoint transition likelihood information is not needed for flow optimization.

Because our approach has a modular design, other data assimilation techniques could be swapped in for SBM approximation and inference depending on computational resources and desired posterior estimation accuracy. Other non-variational inference methods are compatible with state space approximation of SDEs, including particle filter (also known as sequential Monte Carlo) algorithms (Golightly & Kypraios, 2018), stochastic gradient Hamiltonian Monte Carlo (T. Chen et al., 2014), stochastic gradient Langevin dynamics (Brosse et al., 2018), and stochastic gradient Markov chain Monte Carlo (Aicher et al., 2019; Nemeth & Fearnhead, 2021). Emerging multi-model ensemble approaches to data assimilation could include SSM-flow models with explicit representation of nonlinear, multiplicative diffusion structures as a complement to EnKF (Bach & Ghil, 2023). A refined SSM-flow model formulated with nonlinear dynamics and noise could potentially replace multiple models used to implicitly represent nonlinear diffusion, thereby making the ensemble computations more efficient.

Although we opted for mean-field inference with the goal of simplifying interpretation of our results, posterior parameter distributions are often correlated in biogeochemical model outputs (Lu et al., 2017). Future work should therefore focus on more complex $q(\theta)$ variational approximations, including full-rank multivariate logit-normal families that allow for covariance in $\theta$. Such an approach could also mitigate the overconfidence in posterior parameter certainty that resulted from our mean-field approach to variational inference (Kucukelbir et al., 2017; Sujono et al., 2022).

Furthermore, when memory availability inhibits the training of larger neural networks for long $T$ or when another method is faster and more convenient to approximate a particular SBM class, different variational families can be used in place of neural moving average flows for representation and optimization of $q(x|\theta)$. These include multivariate normal distributions with specialized covariance structures (Archer et al., 2015), automatic differentiation VI (Kucukelbir et al., 2017) with Gauss-Markov distributions (Sujono et al., 2022), neural stochastic differential equations (Tzen & Raginsky, 2019; Jia & Benson, 2019; X. Li et al., 2020), and recurrent neural networks (Krishnan et al., 2017; Ryder et al., 2018; Ghosh et al., 2023), among others. Thus, our framework is flexible and can be repurposed as needed for assimilation of different SBMs or Earth system models that vary in complexity and simulation requirements.

Application of our VI approach to model comparison and selection emphasizes the need to further develop goodness-of-fit metrics. Although our SSM-flow results compared favorably with those from SSM-NUTS (Figures 3b, 4), direct quantitative comparison of VI versus MCMC goodness-of-fit is unresolved due to differences in the underlying goals and assumptions of these techniques. Ideally, VI approaches could be quantitatively compared with NUTS, approximate Bayesian computation, integrated nested Laplace approximation, or other Bayesian methods using established metrics such as leave-one-out cross-validation or widely applicable information criterion. Making these comparisons will require new conceptual advances and more analysis of the trade-offs and assumptions underlying different metrics. In the meantime, we used log joint probability as a quantitative comparison metric for different models with the same priors and conditioned on the same data. Looking ahead, Bayesian goodness-of-fit metrics that allow comparisons between VI and other methods are actively being developed (Yao et al., 2018; Giordano et al., 2018), and cross-validation approaches could be potentially integrated with an SSM inference pipeline (Dhaka et al., 2020; Dao et al., 2022), although there may still be limitations due to fundamental differences between Bayesian inference approaches.

Beyond flow architecture design and implementation, another research priority is the further mechanistic refinement, expansion, and development of stochastic SBMs. For example, there is a need for stochastic parameterizations of more complex SBMs with nonlinear drift processes. These models could represent enzyme state variables with drift equations that explicitly simulate nonlinear Michaelis-Menten dynamics. Other model development might include stochastic partial differential equations to represent spatially explicit drift processes. Nonlinear diffusion structures for SBMs could also more realistically reflect the reaction stoichiometry underlying real-world soil processes (Golightly & Wilkinson, 2005, 2011; Fuchs, 2013).

As SBMs with nonlinear drift and diffusion develop further (J. Li et al., 2014; Sulman et al., 2014; Wieder et al., 2015; J. Li et al., 2019; Xie et al., 2020), our flow-based VI framework could be extended to facilitate inference on these models. Because they cannot exactly admit multiplicative diffusion structures, standard implementations of Kalman filters, smoothers, and ensemble derivatives such as EnKF cannot easily accommodate explicit analyses of dynamical systems and specific model parameters associated with non-Gaussian noise (Shen & Tang, 2015; Katzfuss et al., 2016; de Bézenac et al., 2020; X. Wang et al., 2025). Indeed, flow-based methods can outperform Extended and Unscented Kalman filter methods (EKF and UKF) and related particle filter methods in some cases (de Bézenac et al., 2020; X. Chen et al., 2021; Delecki et al., 2023; X. Chen & Li, 2025). Because flows source their model expressivity from invertible bijections (Kong & Chaudhuri, 2020; Köhler et al., 2021; Papamakarios et al., 2021; Zhai et al., 2025), they are not prone to the issue of particle degeneracy, or sample impoverishment, that limits particle filters (T. Li et al., 2014; de Bézenac et al., 2020; X. Chen et al., 2021; X. Wang et al., 2025; X. Chen & Li, 2025). Likewise, flows avoid the linearization inaccuracy that affects EKF and escalating compute costs that scale cubically with state dimensions in UKF.

### 4.3 Conclusion

Our variational inference approach, based on SSM approximation of stochastic SBMs, represents a viable alternative for biogeochemical data assimilation, fitting, and model comparison research. Although additional work is needed to apply this approach with real-world data and complex models, our development of SDE models and deep learning methods for SSM approximation provides a useful foundation for future work. Compared to ODE models, SDE systems provide a more robust framework to accommodate prior densities, initial conditions, and model structures that may be inconsistent with true underlying data-generating processes (Whitaker, 2016; Wiqvist et al., 2021). Moreover, state-space approximation reduces the computational burden of sampling SDE state trajectories for likelihood evaluation. Additionally, the discrete nature of SSMs could facilitate likelihood estimation conditioned on sparsely observed datasets from long-term ecological research, allowing more efficient scaling to larger observation time spans. With a concerted effort to develop more general model comparison metrics and techniques to accommodate sparse and uneven observations, flow-based VI on SSMs could become a valuable tool in the array of data assimilation methods available for inference on biogeochemical models.

## 5 Appendix A

### 5.1 SCON stochastic differential equation model

SDE system equations are frequently written with the state value derivatives $\mathrm{d}x$ on the left-hand side, and consist of a drift coefficient vector, frequently notated as $\alpha$, and a diffusion coefficient matrix, notated as $\beta$, on the right-hand side. For biological SDE models, a square-root diffusion structure is frequently used such that these systems

follow the form

$$\mathrm{d}x_t = \alpha(x_t, t, \theta)\mathrm{d}t + \sqrt{\beta(x_t, t, \theta)}\mathrm{d}W_t \tag{A1}$$

where $\mathrm{d}W_t$ denotes a continuous stochastic Wiener process, or incremental movement along a random walk (Ditlevsen & Samson, 2013). Evolution of SDE trajectories $x$ across a simulation duration $T$ in time increments $\mathrm{d}t$ can be interpreted as a series of small steps whose values are independently drawn from a normal distribution with mean $\alpha(x_t, \theta, t)\mathrm{d}t$ and variance $\beta(x_t, \theta, t)\mathrm{d}t$ (Särkkä & Solin, 2019).

Like the CON model introduced in Allison et al. (2010), SCON has three state dimensions made up of soil organic C (SOC), dissolved organic C (DOC), and microbial biomass C (MBC) densities. We notate total state dimensions with $\mathcal{D}$, so $\mathcal{D} = 3$ for all systems in the SCON family. SOC, DOC, and MBC are respectively notated in the system equations as $S$, $D$, and $M$. Thus, $x_t$, the solutions of the continuous SCON system at time $t$, expand to the vector,

$$x_t = \begin{bmatrix} S_t \\ D_t \\ M_t \end{bmatrix} \tag{A2}$$

and observations of the system $y_t$ are similarly three-dimensional.

We established two SCON versions for inference and data generation use, SCON-C and SCON-SS. SCON-C and SCON-SS share the same underlying $\alpha$ drift vector, equivalent to the deterministic CON dynamics and following the form:

$$\begin{bmatrix} \mathrm{d}S \\ \mathrm{d}D \\ \mathrm{d}M \end{bmatrix} = \begin{bmatrix} \mathcal{I}_S + a_{DS} \cdot k_D \cdot D + a_M \cdot a_{MSC} \cdot k_M \cdot M - k_S \cdot S \\ \mathcal{I}_D + a_{SD} \cdot k_S \cdot S + a_M \cdot (1 - a_{MSC}) \cdot k_M \cdot M - (u_M + k_D) \cdot D \\ u_M \cdot D - k_M \cdot M \end{bmatrix} \mathrm{d}t + \beta^{0.5} \begin{bmatrix} \mathrm{d}W_S \\ \mathrm{d}W_D \\ \mathrm{d}W_M \end{bmatrix} \tag{A3}$$

$\mathcal{I}_S$ and $\mathcal{I}_D$ respectively represent exogenous plant inputs into the soil states of SOC and DOC in units of $\mathrm{mg\,C\,g^{-1}\,soil\,h^{-1}}$. $k_{i\in\{S,D,M\}}$ are linear first-order decay coefficients in units of inverse time (e.g. $\mathrm{h^{-1}}$). $a_{i\in\{DS,SD,M,MSC\}}$ are unitless transfer coefficients that determine the fraction of mass transferred from one state to another (e.g. $a_{DS}$ denotes the fractional transfer from DOC to SOC). $u_M$ is a unitless efficiency coefficient determining the fraction of carbon retained in MBC following DOC uptake. $\beta$ refers to the diffusion matrix component of the SDE and the $W_S$, $W_D$, and $W_M$ elements of the Wiener process vector represent random draws from the distribution $\mathcal{N}(0, \sqrt{\mathrm{d}t})$.

For simplification purposes, the $\beta$ diffusion matrices of both systems were made to be diagonal only and free of covariance diffusion terms. SCON-C diffusion dynamics are given by

$$\beta = \begin{bmatrix} c_S & 0 & 0 \\ 0 & c_D & 0 \\ 0 & 0 & c_M \end{bmatrix} \tag{A4}$$

while SCON-SS diffusion dynamics are

$$\beta = \begin{bmatrix} s_S \cdot S & 0 & 0 \\ 0 & s_D \cdot D & 0 \\ 0 & 0 & s_M \cdot M \end{bmatrix} \tag{A5}$$

Thus, SCON-C diffusion is *additive*, meaning it is independent of the values of states $S$, $D$, and $M$, and also *stationary*, meaning that is not a function of $t$. Meanwhile, SCON-SS noise is *multiplicative*, meaning it is dependent on the states. As such, SCON-C is linear in drift and diffusion, while SCON-SS is linear in drift but nonlinear in diffusion.

We used sinusoidal litter input functions with annual periods that assumed litterfall peaking through late summer and early fall in a pattern resembling those observed in tropical forest ecosystems (Giweta, 2020). The functions are given by

$$\mathcal{I}_{S,t} = 0.001 + 0.0005 \cdot \sin\left(\frac{2\pi}{365 \cdot 24} t\right) \tag{A6}$$

$$\mathcal{I}_{D,t} = 0.0001 + 0.00005 \cdot \sin\left(\frac{2\pi}{365 \cdot 24} t\right) \tag{A7}$$

As was previously instituted for CON (Allison et al., 2010; J. Li et al., 2014), the SCON $k_{i\in\{S,D,M\}}$ decay parameters are temperature sensitive, following the Arrhenius equation (Arrhenius, 1889),

$$k_{i,t} = k_{i,\,\mathrm{ref}} \exp\left[-\frac{Ea_{k_i}}{R}\left(\frac{1}{\mathrm{temp}_t} - \frac{1}{\mathrm{temp}_{\mathrm{ref}}}\right)\right] \tag{A8}$$

where $R$ is the ideal gas constant 8.314 $\mathrm{J\,K^{-1}\,mol^{-1}}$ and $\mathrm{temp}_{\mathrm{ref}}$ specifies a "reference" equilibrium temperature which we set at 283 K.

Through changing values of $k_{i\in\{S,D,M\}}$, SCON systems respond to day-night and seasonal temperature cycles through the composite sinusoid forcing function,

$$\mathrm{temp}_t = \mathrm{temp}_{\mathrm{ref}} + \frac{5t}{80 \cdot 365 \cdot 24} + 10 \cdot \sin\left(\frac{2\pi}{24} t\right) + 10 \cdot \sin\left(\frac{2\pi}{365 \cdot 24} t\right) \tag{A9}$$

The function assumes a gradual linear increase in mean soil surface temperature of 0.0625 $^{\circ}\mathrm{C\ yr^{-1}}$ from the start of the simulation, in line with the upper bound of mean surface temperature increases predicted in emissions scenarios (O'Neill et al., 2017).

SDE systems rarely admit tractable analytic solutions. To sample state trajectories accurately approximating SCON-C and SCON-SS dynamics and construct our synthetic time series data $y$, we used the long-established and reliable Euler-Maruyama SDE solver (Maruyama, 1955) to numerically integrate solution paths $x$ corresponding to $\theta$ randomly sampled from logit-normal distributions. Our solver step size was set to $\mathrm{d}t =$ 0.1 hour. We note that we recover the exact SCON-C and SCON-SS processes in continuous time as $\mathrm{d}t$ is decreased to 0.

If inference involved conditioning with $CO_2$ observations in $y$ in addition to state measurements, model $CO_2$ respiration rate would be computed from the time-corresponding $x$ state values with the equation

$$\mathrm{CO}_{2,\,t} = (1 - a_{SD}) \cdot k_{S,\,t} \cdot S_t + (1 - a_{DS}) \cdot k_{D,\,t} \cdot D_t + (1 - a_M) \cdot k_{M,\,t} \cdot M_t \tag{A10}$$

where $\mathrm{CO}_{2,\,t}$ is in units of $\mathrm{\mu g\,g^{-1}\,soil\,h^{-1}}$. We then sliced $x$ and $CO_2$ time series at some regular interval, i.e. every 1 hour or 5 hours, and normally sampled about the sliced values with an observation error vector $\sigma_{\mathrm{obs}}$ in the manner of

$$y_t \sim \mathcal{N}(x_t, \eta_{\mathrm{obs}}) \tag{A11}$$

to arrive at $y$. We lower bounded $y$ such that $y \in \mathbb{R}_{\geq 0}$ to preclude nonsense negative state measurements. We used constant $\eta_{\mathrm{obs}}$ that was 10% of the overall state mean such that

$$\eta_{\mathrm{obs}} = 0.1 \odot \begin{bmatrix} \bar{S} \\ \bar{D} \\ \bar{M} \end{bmatrix} \tag{A12}$$

where $\odot$ indicates elementwise multiplication. This corresponds to an empirical scenario where measurement instruments and processes introduce a stable level of observation noise. $CO_2$ was similarly observed with noise standard deviation that was 10% of the overall $CO_2$ mean across the total time span in the sampling of $y$ including $CO_2$.

### 5.2 The generalized univariate logit-normal distribution

We used a univariate logit-normal distribution family for our data-generating, informed prior $p(\theta)$, and mean-field variational posterior $q(\theta|y)$ probability density functions. To avoid being restricted to the standard $[0, 1]$ distribution support that the logit-normal density is typically associated with in statistics, we defined a generalized form of the family whose supports could be enclosed between an arbitrary positive $[a, b]$, where $a, b \in \mathbb{R}_{\geq 0}$ and $b > a$. Generalized logit-normal distributions provide similar utility to truncated normal distributions used previously in SBM inference projects for constraining $\theta$ values to finite supports (Xie et al., 2020), but are more stable for backpropagation, as the inverse cumulative distribution function of the truncated normal distribution has inherent stability issues close to support boundaries.

We notate logit-normal distribution parameters in order of desired "target" mean $\mu$, standard deviation $\sigma$, support lower bound $a$, and upper bound $b$ akin to

$$\theta \sim \mathscr{LN}(\mu, \sigma, a, b) \tag{A13}$$

Via passage through a sigmoid function, logit-normal distributions are transformed from normal distributions $\mathscr{N}(\check{\mu}, \check{\sigma})$, where $\check{\mu}$ and $\check{\sigma}$ are respectively the "parent" mean and standard deviation distribution parameters:

$$\check{\theta} \sim \mathscr{N}(\check{\mu}, \check{\sigma}) \tag{A14}$$

$$\theta^{\text{mid}} = \frac{1}{1 + \exp(-\check{\theta})} \tag{A15}$$

$$\theta = (b - a) \cdot \theta^{\text{mid}} + a \tag{A16}$$

The logit-normal distribution has no closed form probability density function and its probability moments are not analytically resolvable, so no formula can be deduced that allows us to make random variable transformations between logit-normal and normal distributions. Hence, to arrive at a particular logit-normal density with target $\mu$ and $\sigma$ in each VI optimization iteration to sample from, we must first numerically solve for the parent $\check{\mu}$ and $\check{\sigma}$ of a normal distribution that corresponds to the desired logit-normal properties following the transformations from equations (A14) to (A16). We can do this with root-finding algorithms like the bisection method that search for an appropriate $\check{\mu}$ in the truncated support interval between $a$ and $b$ and $\check{\sigma}$ within a provided range of tolerated standard deviation values (Daunizeau, 2017).

### 5.3 State space model observation of the SDE

Instead of optimizing SCON $\theta$ via an iterative SDE solver, we optimized time-discretized SSMs approximating the SCON-C and SCON-SS SDEs. SSMs describe the distribution of vectors of discrete observations $y$ conditioned on the distribution of latent states $x$. An observation $y_t$ sampled at time $t$ is

$$y_t \sim p(y_t|x_t) \tag{A17}$$

where $x_t$ is the realization of the latent state at time $t$ constituent to the larger $x$ that depends on the state at the previous time step $x_{t-1}$ and model parameters $\theta$ such that

$$x_t \sim p(x_t|x_{t-1}, \theta) \tag{A18}$$

In other words, $x$ is a Markov chain where transition to subsequent states $x_t$ is conditioned on previous states $x_{t-1}$ and $p(x_t|x_{t-1}, \theta)$ is the *transition density*. To compute and approximate an entire $x$, an initial state $x_0$ must be nominated. $x_0$ can be realized from a probability distribution or fixed. For added randomness and realism, we chose to sample $x_0$ from an established density, $p(x_0)$, which is later accounted for in equation (A25).

The SSM $\theta$ are the same model parameters as in the SDE counterpart. When accounting for the SDE $\alpha$ drift and $\beta$ diffusion dynamics, $x_t$, the latent states of the SSM at time $t$ can be written as

$$x_t = x_{t-1} + \alpha(x_{t-1}, \theta)\Delta t + \epsilon_t \sqrt{\beta(x_{t-1}, \theta)\Delta t} \tag{A19}$$

with the same $\alpha$ and $\beta$ as in (A1). $\epsilon_t$ is a random noise vector of length $\mathcal{D}$ independently sampled via $\epsilon_t \sim \mathcal{N}(0, I_\mathcal{D})$. $I_\mathcal{D}$ is an identity matrix with number of diagonal elements equal to $\mathcal{D}$. $\Delta t$ is the SSM time step, not to be confused with SDE solver time step $\mathrm{d}t$. We used $\Delta t = 1.0$ hour for our SSM approximations in contrast to the aforementioned $\mathrm{d}t = 0.1$ for Euler-Maruyama solving of our data generating processes.

There is overlap between SDEs and SSMs. Both depict the evolution of state values in a series of steps where future values depend on past ones. Both require the specification of initial conditions $x_0$ to compute state trajectories.

However, SDEs and SSMs treat time differently. A key distinction that makes SSM approximation helpful for inference flexibility is that $\Delta t$ can be made relatively large versus SDE solver $\mathrm{d}t$. This is helpful for common biological inference and data assimilation situations where $y$ is sparsely observed due to the expense and difficulty of collecting measurements.

Differential equation systems are instead typically numerically integrated and like SSMs, are solved in discrete steps, as smooth analytic solutions can only be obtained from the simplest systems. But, the differential equation integration procedures still assume that states are evolving in continuous time. The integrating solvers almost always require relatively small integration time steps $\mathrm{d}t$ that are as close to 0 as possible; the solvers tend to fail at higher $\mathrm{d}t$.

The divergent handling of time in SSMs and SDEs renders them more apt for different objectives. SSMs are better suited for estimating observations over long $T$, whereas SDEs are required for precise analyses of accurately simulated system dynamics. With their differing but related purposes, we can ultimately use SSMs to represent discrete observations from continuous SDEs.

### 5.4 VI optimization

Under a Bayesian statistics framework, the goal of statistical inference broadly consists of estimation of the $\theta$ posterior density function for some model, $p(\theta|y)$, conditioned on some data set $y$ via Bayes' rule,

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{p(y)} \tag{A20}$$

$p(y|\theta)$ is the likelihood function, which indicates model goodness-of-fit across various values of individual parameters comprising $\theta$. $p(\theta)$ is the prior probability representing beliefs about $\theta$ uncertainty. $p(y)$ is the probability density of the observed data, sometimes known as the *marginal evidence*.

The prior density $p(\theta)$ can be specified in an informed fashion, as we did in our workflow with distributions that matched our known data-generating distributions (Table S1),

or with less certainty in the absence of empirical information or domain experience. In most cases, $p(y|\theta)$ is not obtainable in closed analytic form and has to be numerically estimated with methods including Monte Carlo sampling and Laplace approximation (Reid, 2015). Assuming proportionality based on (A20), we use

$$p(\theta|y) \propto p(y|\theta)p(\theta) = p(y,\theta) \tag{A21}$$

to estimate $p(\theta|y)$ and practically conduct inference.

For Bayesian inference on SSMs, we account for the transition and observation densities generalized in equations (A18) and (A17), which influence the $\theta$ posterior. Estimation of the posterior of $\theta$ in SSM inference must occur along with estimation of the posterior of $x$, whether in a joint or marginal fashion, in a case such as ours where the transition and observation distributions are not known. We opted for joint estimation. The joint posterior density of $\theta$ and $x$ is notated as $p(\theta, x|y)$. We arrive at an expression for $p(\theta, x|y)$ by substituting (A18) and (A17) into (A21):

$$\begin{aligned} p(\theta, x|y) &\propto p(y, x, \theta) && \text{(A22)}\\ &= p(y|x,\theta)p(x,\theta) && \text{(A23)}\\ &= p(y|x,\theta)p(x|\theta)p(\theta) && \text{(A24)}\\ &= p(x_0)p(\theta)\prod_{i\in\mathfrak{N}} p(y_i|x_i,\theta)\prod_{t=1}^{T} p(x_t|x_{t-1},\theta) && \text{(A25)} \end{aligned}$$

$T$ denotes the final discretized integer time index of $x$. Since we set SSM $\Delta t = 1.0$ hour, our final time index matches total synthetic experiment hours and $T$ can signify both. We also use $T$ to represent the set of $x$ SSM discretization indices not including the initial state at $t = 0$. We can then adopt a $\mathfrak{T} \subseteq T$ to indicate the set of $y$ observation time indices not including an initial observation at $t = 0$, which is required in our VI procedure. The total number of $x$ discretizations is $N = T+1$ when including the $t = 0$ index. $\mathfrak{N} = \{0\} \cup \mathfrak{T}$ notates the full set of $y$ indices.

We note that the $p(y, \theta, x)$ quantity above is referred to as the *joint probability*. In section 3.1, we will describe the use of $\log p(y, \theta, x)$, the *log joint probability*, as an indicator of inference fit quality.

In stochastic VI on SSMs, we optimize a parametric density $q(\theta, x)$ to match the true joint posterior $p(\theta, x|y)$ as closely as possible. Per the probability chain rule, $q(\theta, x)$ expands to,

$$q(\theta, x) = q(x|\theta)q(\theta) \tag{A26}$$

The density functions we select for our marginalized $q(\theta)$ and $q(x|\theta)$ are known as our *variational families*. As mentioned in section 5.2, we chose a mean-field logit-normal variational family for $q(\theta)$ that assumed independent univariate distributions per $\theta$ such that

$$q(\theta) = q(\theta_1, \theta_2, \ldots, \theta_{\mathcal{P}}) = \prod_{j=1}^{\mathcal{P}} q_j(\theta_j) \tag{A27}$$

where $\mathcal{P}$ is the total number of individual SBM $\theta$ and $\mathcal{P} = 14$ for SCON-C and SCON-SS (Table S1). We chose a class of normalizing flow called a *neural moving average flow* in Ryder et al. (2021) for our $q(x|\theta)$ variational family, which we will describe in section 5.5.

We can index individual representatives of our joint variational family by the properties and hyperparameters of the distribution symbolized in aggregate by $\phi_{(\theta,x)}$ such that an instance of $q(\theta, x)$ is notated as $q(\theta, x; \phi_{(\theta,x)})$. $q(\theta, x; \phi_{(\theta,x)})$ can be decomposed

into $q(\theta;\phi_\theta)q(x|\theta;\phi_x)$ since $\phi_{(\theta,x)} = (\phi_\theta, \phi_x)$. $\phi$ are termed *variational parameters*, as they represent the distribution settings that can be varied and tuned to adjust the approximation. For neural network models like flows, variational parameters would include the hidden neural network parameters and weights. A particular distribution can be referred to by its variational parameter index in shorthand.

Our VI framework seeks a set of variational parameters $\phi$ that minimizes discrepancies between $q(\theta, x; \phi_{(\theta,x)})$ and $p(\theta, x|y)$, the approximate and true posteriors. One measure of distance between distributions commonly used in statistics and machine learning literature is the *Kullback-Leibler (KL) divergence* (Kullback & Leibler, 1951; Perez-Cruz, 2008; Joyce, 2011). Targeting the KL divergence $D_{\mathrm{KL}}[q(\theta, x; \phi_{(\theta,x)})||p(\theta, x|y)]$ for minimization, our optimization objective can then be mathematically stated as,

$$q(\theta, x; \phi^*_{(\theta,x)}) = \mathrm{argmin}_{q(\theta,x;\phi_{(\theta,x)})}(D_{\mathrm{KL}}[q(\theta, x; \phi_{(\theta,x)})||p(\theta, x|y)]) \tag{A28}$$

where $\phi^*_{(\theta,x)}$ indexes the set of variational parameters that corresponds to the idealized global KL divergence minimum. After several omitted steps available in Blei et al. (2017), we proceed from (A28) to

$$D_{\mathrm{KL}}[q(\theta, x; \phi_{(\theta,x)})||p(\theta, x|y)] = \mathbb{E}_{q(\theta,x;\phi_{(\theta,x)})}[\log q(\theta, x; \phi_{(\theta,x)})] - \mathbb{E}_{q(\theta,x;\phi_{(\theta,x)})}[\log p(y|\theta, x)] \tag{A29}$$

$$= \mathbb{E}_{q(\theta,x;\phi_{(\theta,x)})}[\log q(\theta, x; \phi_{(\theta,x)})] - \mathbb{E}_{q(\theta,x;\phi_{(\theta,x)})}[\log p(y, \theta, x)] + \log p(y) \tag{A30}$$

where the expectations $\mathbb{E}$ subscripted with $q(\theta, x; \phi_{(\theta,x)})$ are taken with respect to the density $q(\theta, x; \phi_{(\theta,x)})$.

Reviewing the notion that $p(y)$ and in turn, the log marginal evidence, are typically unavailable (Christensen et al., 2010), we then rely on a reduced and rearranged expression that constitutes the ELBO function, notated as $\mathcal{L}$,

$$\mathcal{L}[\phi_{(\theta,x)}] = \mathbb{E}_{q(\theta,x;\phi_{(\theta,x)})}[\log p(y, \theta, x)] - \mathbb{E}_{q(\theta,x;\phi_{(\theta,x)})}[\log q(\theta, x; \phi_{(\theta,x)})] \tag{A31}$$

$$= \mathbb{E}_{q(\theta,x;\phi_{(\theta,x)})}[\log p(y, \theta, x) - \log q(\theta, x; \phi_{(\theta,x)})] \tag{A32}$$

$$= \mathbb{E}_{q(\theta,x;\phi_{(\theta,x)})}\left\langle \log p(y, \theta, x) - \log[q(x|\theta;\phi_x)q(\theta;\phi_\theta)]\right\rangle \tag{A33}$$

$$= \mathbb{E}_{q(\theta,x;\phi_{(\theta,x)})}\left\langle \log p(y, \theta, x) - \log q(x|\theta;\phi_x) - \log q(\theta;\phi_\theta)\right\rangle \tag{A34}$$

Substituting in (A25) for $p(y, \theta, x)$ results in

$$\mathcal{L}[\phi_{(\theta,x)}] = \mathbb{E}_{q(\theta,x;\phi_{(\theta,x)})}\langle \log \left[p(x_0)p(\theta)\prod_{i\in\mathfrak{N}} p(y_i|x_i,\theta)\prod_{t=1}^{T} p(x_t|x_{t-1},\theta)\right] - \log q(x|\theta;\phi_x) - \log q(\theta;\phi_\theta)\rangle \tag{A35}$$

which, recalling that the set of total $y$ indices $\mathfrak{N} = \{0\} \cup \mathfrak{T}$, expands into

$$\mathcal{L}[\phi_{(\theta,x)}] = \mathbb{E}_{q(\theta,x;\phi_{(\theta,x)})}\langle \log p(x_0) + \log p(\theta) + \log p(y_0|x_0,\theta) + \sum_{i\in\mathfrak{T}} \log p(y_i|x_i,\theta) + \sum_{t=1}^{T} \log p(x_t|x_{t-1},\theta) - \log q(x|\theta;\phi_x) - \log q(\theta;\phi_\theta)\rangle \tag{A36}$$

We will decompose the marginal variational log-density of $x$, $\log q(x|\theta;\phi_x)$, in more granular detail when we describe the architecture of the neural moving average flow in section 5.5.

Following from equation (A30), the ELBO function is called as such because it is the "evidence lower bound" of the log marginal evidence:

$$\log p(y) = \mathcal{L}[\phi_{(\theta,x)}] + D_{\mathrm{KL}}[q(\theta, x; \phi_{(\theta,x)})||p(\theta, x|y)] \tag{A37}$$

$$\geq \mathcal{L}[\phi_{(\theta,x)}] \tag{A38}$$

Maximizing $\mathcal{L}[\phi_{(\theta,x)}]$, or minimizing the negative ELBO $-\mathcal{L}$, as we need to do in machine learning libraries like PyTorch that implement gradient descent rather than ascent, is commensurate to minimizing $D_{\mathrm{KL}}[q(\theta, x; \phi_{(\theta,x)})||p(\theta, x|y)]$. Hence, $\mathcal{L}[\phi_{(\theta,x)}]$ is our optimization objective function.

For pithier description of the ELBO gradient, $\nabla\mathcal{L}$, used to update $\phi_{(\theta,x)}$ via automatic differentiation, we set $\log p(y, \theta, x) - \log q(\theta, x; \phi_{(\theta,x)})$ in (A32) equal to $\mathscr{R}(\theta, x, y, \phi)$, where $\mathscr{R}$ is a log-density ratio function. This reduces the ELBO equation to

$$\mathcal{L}[\phi_{(\theta,x)}] = \mathbb{E}_{q(\theta,x;\phi_{(\theta,x)})}\left[\mathscr{R}(\theta, x, y, \phi_{(\theta,x)})\right] \tag{A39}$$

and the ELBO gradient is

$$\nabla\mathcal{L}[\phi_{(\theta,x)}] = \nabla_\phi \left\langle \mathbb{E}_{q(\theta,x;\phi_{(\theta,x)})}\left[\mathscr{R}(\theta, x, y, \phi_{(\theta,x)})\right] \right\rangle \tag{A40}$$

$$= \nabla_\phi \left[ \int_\theta \int_x q(\theta, x; \phi_{(\theta,x)}) \mathscr{R}(\theta, x, y, \phi_{(\theta,x)}) \mathrm{d}x\mathrm{d}\theta \right] \tag{A41}$$

$$= \int_\theta \int_x \nabla_\phi \left[ q(\theta, x; \phi_{(\theta,x)}) \mathscr{R}(\theta, x, y, \phi_{(\theta,x)}) \right] \mathrm{d}x\mathrm{d}\theta \tag{A42}$$

which decomposes to

$$\nabla\mathcal{L}[\phi_{(\theta,x)})] = \int_\theta \int_x q(\theta, x; \phi_{(\theta,x)}) \nabla_\phi \left[\mathscr{R}(\theta, x, y, \phi_{(\theta,x)})\right] \mathrm{d}x\mathrm{d}\theta + \int_\theta \int_x \mathscr{R}(\theta, x, y, \phi_{(\theta,x)}) \nabla_\phi \left[q(\theta, x; \phi_{(\theta,x)})\right] \mathrm{d}x\mathrm{d}\theta \tag{A43}$$

Note that the gradients $\nabla_\phi$ are taken with respect to the variational parameters. This presents a complication, as examining the second term of (A43), we are left with the situation that $\nabla_\phi \left[q(\theta, x; \phi_{(\theta,x)})\right]$ is by and large unavailable, as $q$ can be sampled from, but is usually not analytically differentiable. Our joint variational family $q$ is no exception to that pattern; our marginal mean-field $q(\theta; \phi_\theta)$ has the straightforward analytic form given in (A27), but use of the neural moving average flow as the variational family for $q(x|\theta; \phi_x)$ precludes the overall joint density $q(\theta, x; \phi_{(\theta,x)})$ from having an orderly closed form.

To ultimately compute the gradient of an expectation as in (A40) in numerical fashion, we thereby turn to the *reparameterization trick* set forth in Salimans and Knowles (2013) and Kingma and Welling (2014). The reparameterization trick involves setting $(\theta, x)$ as an output of an invertible, deterministic, and differentiable function $g(z, \phi_{(\theta,x)})$, where $z$ is a random vector sampled from some fixed density $q(z)$. This enables us to tractably take a gradient of a simpler fixed distribution whose probability density function is easier to differentiate and not dependent on the variational parameters $\phi$ (Ruiz et al., 2016).

In our case, $z$ elements are sampled from standard normal distributions and undergo invertible transformations to proceed to $x$. $\theta$ is still directly sampled from its mean-field logit-normal family described in section 5.4 as part of the operations of $g$. Hence, $\mathcal{L}$ can be estimated with each VI training iteration with Monte Carlo sampling of $z$ and

$\theta$ entries starting with the steps

$$z^{(s)} \sim \mathcal{N}(0, I_N) \tag{A44}$$

$$\theta^{(s)}, x^{(s)} = g(z^{(s)}, \phi_{(\theta,x)}) \tag{A45}$$

where $I_N$ is an identity matrix with number of diagonal entries matching the total $x$ discretization indices $N$. The superscript $(s)$ indexes an individual draw from a distribution. We can then re-frame (A40) from an analytically intractable gradient of an expectation to a numerically assessable expectation of a gradient with

$$\nabla\mathcal{L}[\phi_{(\theta,x)}] = \nabla_\phi \left\langle \mathbb{E}_{q(z)} \left[ \mathscr{R}(\theta, x, y, \phi_{(\theta,x)}) \right] \right\rangle \tag{A46}$$

$$= \mathbb{E}_{q(z)} \left[ \nabla_\phi \left\langle \mathscr{R}(\theta, x, y, \phi_{(\theta,x)}) \right\rangle \right] \tag{A47}$$

$$\approx \frac{1}{\mathcal{S}} \sum_{s=1}^{\mathcal{S}} \nabla_\phi \left\langle \mathscr{R}(\theta^{(s)}, x^{(s)}, y, \phi_{(\theta,x)}) \right\rangle \tag{A48}$$

$$\widehat{\nabla\mathcal{L}[\phi_{(\theta,x)}]} = \frac{1}{\mathcal{S}} \sum_{s=1}^{\mathcal{S}} \nabla_\phi \left\langle \mathscr{R} \left[ g(z^{(s)}, \phi_{(\theta,x)}), y, \phi_{(\theta,x)}) \right] \right\rangle \tag{A49}$$

$\mathcal{S}$ denotes the total number of independent $\theta$ and $z$ samples drawn per training iteration. $\widehat{\nabla\mathcal{L}[\phi_{(\theta,x)}]}$ specifies the Monte Carlo estimate of $\nabla\mathcal{L}[\phi_{(\theta,x)}]$.

### 5.5 Masked neural moving average flow architecture

In general, flow architecture includes a chain of *bijections*, or invertible transformation functions mapping an object in a set one-to-one to an object in another set. Flows can be decomposed into

$$x = g(z) = g_M \circ g_{M-1} \circ \cdots \circ g_m \circ \cdots \circ g_1(z) \tag{A50}$$

where $\circ$ notates function composition operations and $M$ marks the total number of bijections. In the generative direction, our flow takes us from a random object $z$, as defined in section 5.4, to a random object $x$ corresponding to a set of approximated SCON trajectories in the SSM.

A generative flow is constructed such that computation of $\log q(x|\theta;\phi_x)$ in (A36) is available to facilitate the optimization of $q(x|\theta;\phi_x)$. The log-density of $x$ is available through the change of variables formula:

$$\log q(x) = \sum_{t=1}^{T} \varphi(z_t) - \log|\det J| \tag{A51}$$

$$\log q(x) = \sum_{t=1}^{T} \varphi(z_t) - \log \prod_{m=1}^{M} |\det J_m| \tag{A52}$$

$$\log q(x) = \sum_{t=1}^{T} \varphi(z_t) - \sum_{m=1}^{M} \log|\det J_m| \tag{A53}$$

where $J$ is the Jacobian matrix of the overall transformation and $J_m$ is the Jacobian of bijection $g_m$ with respect to the intermediate transformed variable $g_{m-1} \circ g_{m-2} \circ \cdots \circ g_1(z)$. We use $\varphi(z_t)$ to indicate the log-density of each element of $z$, so each $\varphi(z_t)$ is then a unit standard normal log-density in our framework. We notate the density function of $z$ with $q(z)$. Since $q(z)$ is the starting distribution before transformations are layered, it is also termed the *base distribution*.

The particular flow we implemented as the marginal variational family for $q(x|\theta)$ was patterned after the original neural moving average flow introduced in Ryder et al.

(2021). Neural moving average flows include *affine* bijections (Dinh et al., 2015; Kingma et al., 2016; Dinh et al., 2017; Papamakarios et al., 2017) among the functions constituting $g$ in which an $x^{\text{out}}$ is transformed from an $x^{\text{in}}$ in the general form of

$$x^{\text{out}} = \mu + \sigma \odot x^{\text{in}} \tag{A54}$$

where $\odot$ represents elementwise multiplication to denote that $\mu$, $\sigma$, and $x^{\text{in}}$ can be matrices and vectors in addition to scalars, though our explicit situation involves scalars. $\mu$ and $\sigma$ are respectively known as shift and scale values of the bijection and it is required that $\sigma \in \mathbb{R}_+$. Cumulative $\mu$ and $\sigma$ values of a flow are usually implemented as trained outputs of a neural network and are super- and subscripted to identify the transformation layer and input elements they correspond to. They are notated as such by convention and not to be confused with the similarly notated target logit-normal mean and standard deviation parameters in section 5.2.

These linear affine transformations are basic in structure and consequently are individually not so *expressive*, or able to flexibly transition a base distribution into substantially different distributions of varying complexity. However, when layered repeatedly and stacked, their cumulative expressivity increases and with sufficient layers, composite affine functions can come to embody any distribution that is log-concave and bookended by declining density tails (Lee et al., 2021), which represents a large swath of probability distributions.

Neural moving average flows are specifically distinguished from other flows containing affine layers through their execution of affine bijections with 1-dimensional convolutional neural networks (CNNs). To apply 1-dimensional CNNs rather than 2-dimensional CNNs, we note that for systems with $\mathcal{D} > 1$, like SCON family instances, we must begin with $z$ in a 1-dimensional "melted" form that is $\mathcal{D}{\cdot}T$ elements in length before reshaping the final transformed $x$ to a $\mathcal{D} \times T$ matrix matching the SDE solution structure demonstrated in (A2) following the conclusion of $g$. Thus, in equations (A44) and (A53), we respectively replace $N$ and $T$ in each equation with $\mathcal{D} \cdot T$ in our implementation.

Through *masking*, in which inputs to the convolution patch are zeroed out through multiplication by weights, the flow is imbued with an *autoregression* property in which the $\sigma_i$ and $\mu_i$ values producing an $i$th element of an output vector $x$ does not depend and convolve on any element $z_{j \geq i}$ in the base input vector (Papamakarios et al., 2017). This autoregression is critical for the intent of arranging the computation of $\sum_{m=1}^{M} \log |\det J_m|$ in (A53) to be manageable. The autoregression ensures that $J$ is a diagonal matrix whose non-zero elements are the $\sigma$ scale parameters underlying the overall transformation, which simplifies calculation of $\det J$ and $\log q(x)$ to

$$\log q(x) = \sum_{t=1}^{T} \varphi(z_t) - \sum_{m=1}^{M} \sum_{t=1}^{T} \log \sigma_t^m \tag{A55}$$

where $\sigma_t^m$ is the scale parameter associated with the bijection or transformation producing the $t$th term of the $m$th affine layer. Masking ensures that when calculating a new value at a specific position, only the previous values in the sequence are considered, which is crucial to maintain the invertibility of the flow and allow efficient density estimation.

Autoregressive convolutions and affine bijections used in our flow implementation occur within *residual blocks*. These blocks contain layers that mitigate training and approximation errors as network depth increases. Residual blocks do this with the use of *skip connections*, which carry the output from previous layers to serve as input to subsequent layers and, in doing so, prevent noisy degradation of the information cascading through the network (He et al., 2016).

In each residual block, we perform two masked 1-dimensional convolutions, Convolution A and Convolution B, that each have a kernel length of 3 elements and a stride length of 1. To enshrine autoregressiveness of the flow, Convolution A applies a kernel masked as $[1, 0, 0]$ that outputs a shift and scale value pair. The Convolution A operation and associated affine bijection can be generally expressed as

$$(\mu_i, \sigma_i) = f_i^{\mathrm{A}}(x_{i-1}^{\mathrm{in}}) \tag{A56}$$

$$x_i^{\mathrm{out}} = \mu_i + \sigma_i \cdot x_i^{\mathrm{in}} \tag{A57}$$

where $\mu_i$, $\sigma_i$, $x_i^{\mathrm{in}}$, and $x_i^{\mathrm{out}}$ are scalar elements of vectors and $f_i^{\mathrm{A}}$ is the Convolution A operation. The subsequent Convolution B involves a single stride kernel masked as $[1, 1, 0]$ and it can be expressed together with its associated bijection as

$$(\mu_i, \sigma_i) = f_i^{\mathrm{B}}(x_{i-1}^{\mathrm{in}}, x_i^{\mathrm{in}}) \tag{A58}$$

$$x_i^{\mathrm{out}} = \mu_i + \sigma_i \cdot x_i^{\mathrm{in}} \tag{A59}$$

Combined, the two convolutions in sequence have a total receptive field length of 2.

To be able to produce the $\mu$ and $\sigma$ parameters associated with the affine transformation of vector endpoint elements under autoregressive alignment, both convolutions require *zero padding*, in which zero elements are added to either end of the vector. Without zero padding, the kernels producing $(\mu_1^{\mathrm{mid}}, \sigma_1^{\mathrm{mid}})$ and $(\mu_1^{\mathrm{out}}, \sigma_1^{\mathrm{out}})$ would lack 1 element to convolve on, and the kernels sourcing $(\mu_N^{\mathrm{mid}}, \sigma_N^{\mathrm{mid}})$ and $(\mu_N^{\mathrm{out}}, \sigma_N^{\mathrm{out}})$ would by overhang their vectors by 1. A zero pad of length 1 was thereby sufficient for our purposes.

In our convolution implementation, input is expanded into many *channels*, which are duplicates of the input vector that are stacked on top of each other in a matrix. At each convolution stage, the same kernel is applied in parallel across all the channels. Enlarging channel depth broadens the space of neural network weight values constituting $f_i^{\mathrm{A}}$ and $f_i^{\mathrm{B}}$ that can be explored per training iteration. Following machine learning convention, we set the number of channels at 96 for both convolutions and did not experiment further with channel depth. Our implementation also uses auxiliary features extracted from $y$ and observation indices $\mathfrak{N}$ in the form of vectors stacked on top of the input channels to inform training of the neural network weights associated with the shift and scale values. Further elaboration on the incorporation of auxiliary information is available in the supplement of Ryder et al. (2021).

In the overall flow procedure, the convolutions and affine bijections in the affine residual block are linked with other transformations that we organize into repeatable sets of layers. Preceding the affine blocks are *order-reversing permutations*, in which element order of a vector input is flipped such that a vector $[x_1^{\mathrm{in}}, x_2^{\mathrm{in}}, ..., x_N^{\mathrm{in}}]$ becomes $[x_1^{\mathrm{out}}, x_2^{\mathrm{out}}, ..., x_N^{\mathrm{out}}] = [x_N^{\mathrm{in}}, x_{N-1}^{\mathrm{in}}, ..., x_1^{\mathrm{in}}]$. Order-reversing permutations are a method of extending the expressivity and stability of a flow by enabling more complex dependency structures while preserving flow autoregression (Papamakarios et al., 2021). We found that adding order reversals allowed us to modestly boost our ELBO learning rates. The permutations can be seamlessly interspersed between other transformations since their absolute Jacobian determinant is valued at 1, so they do not affect the computation of $\log q(x)$.

Differing from the neural moving average flow of Ryder et al. (2021), our flow follows affine blocks with *batch renormalization* transformations. Batch renormalization is a simple extension of *batch normalization*, which is a means of normalizing and regularizing our variational samples such that our optimization is less influenced by random fluctuations in neural network weights and sample characteristics from one training iteration to the next (Ioffe & Szegedy, 2015). Similar in intent to permutations, batch normalization and renormalization are applied to bolster algorithm stability and flexibility with increasing layer depth. They empirically allow VI algorithms to tolerate higher

learning rates (Bjorck et al., 2018), poor initialization of variational parameters $\phi$ (Zhu et al., 2020), and erratic base distribution $z^{(s)}$ draws.

Batch normalization and renormalization overlap in the following steps that compute a *batch mean* $\mu_{\mathcal{S}}$ and *batch standard deviation* $\sigma_{\mathcal{S}}$ from input $x^{\text{in}}$ samples, not to be confused with the affine bijection and logit-normal $\mu$ and $\sigma$:

$$\mu_{\mathcal{S}} = \frac{1}{\mathcal{S}} \sum_{s=1}^{\mathcal{S}} x_s^{\text{in}} \tag{A60}$$

$$\sigma_{\mathcal{S}} = \sqrt{\varepsilon + \frac{1}{\mathcal{S}} \sum_{s=1}^{\mathcal{S}} (x_s^{\text{in}} - \mu_{\mathcal{S}})^2} \tag{A61}$$

where $\varepsilon$ is a small constant added for stability. $\mu_{\mathcal{S}}$ and $\sigma_{\mathcal{S}}$ are involved in computation of the optimization objective—again, $\mathcal{L}[\phi_{(\theta,x)}]$ for our purposes—during the model training phase. They also update a lagging *running average* $\mu_{\mathcal{R}}$ and *running mean* $\sigma_{\mathcal{R}}$ that are less sensitive to change. $\mu_{\mathcal{R}}$ and $\sigma_{\mathcal{R}}$ are used after training of the model—the joint variational family $q(\theta, x; \phi_{(\theta,x)})$ in this setting—has been halted to estimate the objective metric at the testing stage.

In the testing phase, batch renormalization and normalization are equivalent in transforming input to output:

$$x^{\text{mid}} = \frac{x^{\text{in}} - \mu_{\mathcal{R}}}{\sigma_{\mathcal{R}}} \tag{A62}$$

$$x^{\text{out}} = \gamma \cdot x^{\text{mid}} + \Upsilon \tag{A63}$$

The collection of $\gamma_t^m$ and $\Upsilon_t^m$ parameters in each flow layer set are learned neural network outputs. Batch renormalization diverges from batch normalization during training with the steps

$$r = \frac{\sigma_{\mathcal{S}}}{\sigma_{\mathcal{R}}} \tag{A64}$$

$$d = \frac{\mu_{\mathcal{S}} - \mu_{\mathcal{R}}}{\sigma_{\mathcal{R}}} \tag{A65}$$

$$x^{\text{mid}} = \frac{x^{\text{in}} - \mu_{\mathcal{S}}}{\sigma_{\mathcal{S}}} \cdot r + d \tag{A66}$$

$$x^{\text{out}} = \gamma \cdot x^{\text{mid}} + \Upsilon \tag{A67}$$

where $r$ and $d$ are variable correction factors. $r$ and $d$ are intended to limit the divergence between batch and running sample characteristics. $r$ is clipped between the interval $[1/r_{\max}, r_{\max}]$, where $r_{\max}$ is increased to 3 over the course of inference, and $d$ is clipped between the interval $[-d_{\max}, d_{\max}]$, where $d_{\max}$ is increased to 5. These intervals were established based on guidelines from previous empirical work (Ioffe, 2017). Batch normalization is a special case of batch renormalization where $r = 1$ and $d = 0$.

Batch renormalization's changes more tightly correlate the batch and running sample characteristics and have been documented to minimize discrepancy between train and test objectives (Ioffe, 2017). We observed this with our ELBO results, where consistent gaps remained between the train and test $\mathcal{L}[\phi_{(\theta,x)}]$ until we swapped batch normalization for renormalization. Batch renormalization also improves training on low batch sizes (Ioffe, 2017; Summers & Dinneen, 2020), and in our position where variational path samples were limited by GPU video memory constraints, renormalization was helpful for decreasing the total number of training iterations we needed for algorithm convergence. Note that in VI, convergence corresponds to evidence of training of the model toward an optimization objective minimum for gradient descent (or maximum for gradient ascent),

whereas convergence in MCMC corresponds to evidence of algorithm approach toward similar posterior density estimates across multiple chains.

With batch (re)normalization layers, $\log q(x)$ accrues log determinant Jacobian summation terms corresponding to those transformations and develops from (A55) to become, in the training phase,

$$\log q(x) = \sum_{t=1}^{T} \varphi(z_t) - \sum_{m=1}^{M} \sum_{t=1}^{T} \left[\log \sigma_t^m - \log r_t^m - \log \gamma_t^m + \log \sigma_{\mathcal{S},t}^m\right] \tag{A68}$$

or in the testing phase,

$$\log q(x) = \sum_{t=1}^{T} \varphi(z_t) - \sum_{m=1}^{M} \sum_{t=1}^{T} \left[\log \sigma_t^m - \log \gamma_t^m + \log \sigma_{\mathcal{R},t}^m\right] \tag{A69}$$

where we now take $M$ to mark the total number of layer sets rather than layers as we did before in (A55). This assumes that each layer set always includes 1 single affine block and 1 batch renormalization layer. Substituting (A68) or (A69) into (A36) for $\log q(x|\theta;\phi_x)$ leads respectively to our fully decomposed train or test $\mathcal{L}[\phi_{(\theta,x)}]$ calculation unless an optional single softplus transformation is used to ensure constraint of flow output to $\mathbb{R}_{\geq 0}$. In that case, the resulting train $\log q(x)$ is

$$\log q(x) = \sum_{t=1}^{T} \varphi(z_t) - \sum_{m=1}^{M} \sum_{t=1}^{T} \left[\log \sigma_t^m - \log r_t^m - \log \gamma_t^m + \log \sigma_{\mathcal{S},t}^m\right] - \sum_{t=1}^{T} \log(-\mathrm{e}^{-x_t} + 1) \tag{A70}$$

where $x$ is our terminally transformed random variable following softplus constraint. Setting

$$\lambda_t = \varphi(z_t) - \log(-\mathrm{e}^{-x_t} + 1) - \sum_{m=1}^{M} \left(\log \sigma_t^m - \log r_t^m - \log \gamma_t^m + \log \sigma_{\mathcal{S},t}^m\right) \tag{A71}$$

$$\log q(x) = \sum_{t=1}^{T} \lambda_t \tag{A72}$$

our fully decomposed train $\mathcal{L}[\phi_{(\theta,x)}]$ calculation that we use in each iteration of VI optimization (Algorithm 1) then consolidates from (A36) into

$$\mathcal{L}[\phi_{(\theta,x)}] = \mathbb{E}_{q(\theta,x;\phi_{(\theta,x)})}\langle \log p(\theta) + \log p(y_0|x_0,\theta) - \log q(\theta;\phi_\theta) + \sum_{i\in\mathfrak{T}} \log p(y_i|x_i,\theta) + \sum_{t=1}^{T} \left[\log p(x_t|x_{t-1},\theta) - \lambda_t\right]\rangle \tag{A73}$$

with softplus flow termination. The test ELBO equation is equivalent except for use of a different $\lambda_t$ assignment that lacks the $\log r_t^m$ term and swaps $\sigma_{\mathcal{S},t}^m$ for $\sigma_{\mathcal{R},t}^m$.

It is apparent that each layer set of our neural moving average flow corresponds to a matrix of hidden parameters, including affine and batch renormalization parameters, of dimensions $[T, h]$, where $h$ is the count of hidden parameters per layer set. Thus, when conditioning on long, dense $T$ data that is complex in such a manner that would require many layer sets for flow representation, we note that a different choice of marginal variational family for $q(x|\theta)$ aside from the neural moving average flow may be appropriate for minimizing computational expense.

### 5.6 Kalman smoother

The Kalman smoother procedure is a two part process consisting of a *forward pass* followed by a *backward pass*. The forward pass computes a "filtering" posterior $p(x_t|y_{0:t})$, which notates the posterior of $x_t$ given observations up to the time indexed by $t$, going forward in time from $t = \{0, \ldots, T\}$. The backward pass computes a "smoothing" posterior $p(x_t|y)$, which notates the posterior of $x_t$ given all observations, going backward in time from $t = \{T, \ldots, 0\}$. Reconciling the "filtering" and "smoothing" posteriors produces the true $p(x|y)$. Flow VI in contrast can only numerically estimate $p(x|y)$ through a variational approximation, and can function without exact knowledge of $\theta$ given uninformed prior distributions, estimating the joint density $p(x, \theta|y)$ via variational approximations.

## Acknowledgments

We thank Anton Obukhov (ETH Zurich) for his PyTorch truncated normal distribution class that was used in exploratory work that this study was built on and Andrew Golightly (Durham University) for his advice on ODE-to-SDE conversion and reparameterization. We thank the anonymous referees for their time, attention, and suggestions that were instrumental for improving this manuscript.

This research was financially supported by the U.S. National Science Foundation (grant no. DEB-1900885 and IIS-1816365), the U.S. Department of Energy (grant no. DE-SC0014374 and DE-SC0020382), and a UC Irvine ICS Exploration Research Award.

The authors affirm that they have no financial conflict of interest.

## 6 Open Research

A repository is available online (Xie, 2024) containing synthetic time series data of soil pool state and $CO_2$ observations, Python notebooks containing the code for SCON-C and SCON-SS data-generating processes, and Python modules scaffolding the neural moving average flow VI algorithm.

## References


Abs, E., Leman, H., & Ferrière, R. (2020). A multi-scale eco-evolutionary model of cooperation reveals how microbial adaptation influences soil decomposition. *Communications Biology*, *3*(1). Retrieved from https://doi.org/10.1038/s42003-020-01198-4 doi: 10.1038/s42003-020-01198-4

Aicher, C., Ma, Y.-A., Foti, N. J., & Fox, E. B. (2019). Stochastic Gradient MCMC for State Space Models. *SIAM Journal on Mathematics of Data Science*, *1*(3). Retrieved from https://doi.org/10.1137/18M1214780 doi: 10.1137/18M1214780

Allison, S. D., Wallenstein, M. D., & Bradford, M. A. (2010). Soil-carbon response to warming dependent on microbial physiology. *Nature Geoscience*, *3*(5). Retrieved from https://doi.org/10.1038/ngeo846 doi: 10.1038/ngeo846

*Anaconda Software Distribution.* (2020). Anaconda Inc. Retrieved 2022-06-17, from https://docs.anaconda.com/

Archer, E., Park, I. M., Buesing, L., Cunningham, J., & Paninski, L. (2015). *Black box variational inference for state space models.* arXiv. Retrieved from https://arxiv.org/abs/1511.07367 doi: 10.48550/ARXIV.1511.07367

Arrhenius, S. (1889). Über die Dissociationswärme und den Einfluss der Temperatur auf den Dissociationsgrad der Elektrolyte. *Zeitschrift für Physikalische Chemie*, *4U*(1). Retrieved from https://doi.org/10.1515/zpch-1889-0408 doi: 10.1515/zpch-1889-0408

Bach, E., & Ghil, M. (2023). A Multi-Model Ensemble Kalman Filter for Data Assimilation and Forecasting. *Journal of Advances in Modeling Earth Systems*, *15*(1), e2022MS003123. Retrieved 2025-04-09, from https://onlinelibrary.wiley.com/doi/abs/10.1029/2022MS003123 (_eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/2022MS003123) doi: 10.1029/2022MS003123

Betancourt, M. (2017). *A Conceptual Introduction to Hamiltonian Monte Carlo.* arXiv. Retrieved from http://arxiv.org/abs/1701.02434

Bjorck, N., Gomes, C. P., Selman, B., & Weinberger, K. Q. (2018). Understanding Batch Normalization. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems* (Vol. 31). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper/2018/file/36072923bfc3cf47745d704feb489480-Paper.pdf

Blei, D. M., Kucukelbir, A., & McAuliffe, J. D. (2017). Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, *112*(518). doi: 10.1080/01621459.2017.1285773

Bradford, M. A., Wood, S. A., Addicott, E. T., Fenichel, E. P., Fields, N., González-Rivero, J., ... Wieder, W. R. (2021). Quantifying microbial control of soil organic matter dynamics at macrosystem scales. *Biogeochemistry*, *156*(1). Retrieved from https://doi.org/10.1007/s10533-021-00789-5 doi: 10.1007/s10533-021-00789-5

Brosse, N., Durmus, A., & Moulines, E. (2018). The promises and pitfalls of Stochastic Gradient Langevin Dynamics. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems* (Vol. 31). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper/2018/file/335cd1b90bfa4ee70b39d08a4ae0cf2d-Paper.pdf

Browning, A. P., Warne, D. J., Burrage, K., Baker, R. E., & Simpson, M. J. (2020). Identifiability analysis for stochastic differential equation models in systems biology. *Journal of The Royal Society Interface*, *17*(173). Retrieved from https://royalsocietypublishing.org/doi/abs/10.1098/rsif.2020.0652 doi: 10.1098/rsif.2020.0652

Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., ... Riddell, A. (2017). Stan: A probabilistic programming language. *Journal of Statistical Software*, *76*(1). doi: 10.18637/jss.v076.i01

Chen, R. T. Q., Rubanova, Y., Bettencourt, J., & Duvenaud, D. (2018). Neural ordinary differential equations. In *Proceedings of the 32nd international conference on neural information processing systems* (p. 6572–6583). Red Hook, NY, USA: Curran Associates Inc.

Chen, T., Fox, E., & Guestrin, C. (2014). Stochastic Gradient Hamiltonian Monte Carlo. In E. P. Xing & T. Jebara (Eds.), *Proceedings of the 31st International Conference on Machine Learning* (Vol. 32). Beijing, China: PMLR. Retrieved from `https://proceedings.mlr.press/v32/cheni14.html`

Chen, X., & Li, Y. (2025). *Normalizing Flow-based Differentiable Particle Filters.* Retrieved from `https://arxiv.org/abs/2403.01499`

Chen, X., Wen, H., & Li, Y. (2021). Differentiable Particle Filters through Conditional Normalizing Flow. In *2021 IEEE 24th International Conference on Information Fusion (FUSION)* (p. 1-6). doi: 10.23919/FUSION49465.2021.9626998

Christensen, R., Johnson, W., Branscum, A., & Hanson, T. E. (2010). *Bayesian Ideas and Data Analysis: An Introduction for Scientists and Statisticians* (1st ed.). Taylor & Francis.

Cobb, A. D. (2023). hamiltorch: A pytorch-based library for hamiltonian monte carlo. In *Proceedings of cyber-physical systems and internet of things week 2023* (p. 114–115). New York, NY, USA: Association for Computing Machinery. Retrieved from `https://doi.org/10.1145/3576914.3587528` doi: 10.1145/3576914.3587528

Dao, V. H., Gunawan, D., Tran, M.-N., Kohn, R., Hawkins, G. E., & Brown, S. D. (2022). Efficient selection between hierarchical cognitive models: Cross-validation with variational Bayes. *Psychological Methods*. doi: 10.1037/met0000458

Daunizeau, J. (2017). *Semi-analytical approximations to statistical moments of sigmoid and softmax mappings of normal variables.* arXiv. Retrieved from `https://arxiv.org/abs/1703.00091` doi: 10.48550/ARXIV.1703.00091

de Bézenac, E., Rangapuram, S. S., Benidis, K., Bohlke-Schneider, M., Kurle, R., Stella, L., ... Januschowski, T. (2020). Normalizing Kalman Filters for Multivariate Time Series Analysis. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, & H. Lin (Eds.), *Advances in Neural Information Processing Systems* (Vol. 33, pp. 2995–3007). Curran Associates, Inc. Retrieved from `https://proceedings.neurips.cc/paper_files/paper/2020/file/1f47cef5e38c952f94c5d61726027439-Paper.pdf`

Delecki, H., Kruse, L. A., Schlichting, M. R., & Kochenderfer, M. J. (2023). Deep normalizing flows for state estimation. In *2023 26th international conference on information fusion (fusion)* (p. 1-6). doi: 10.23919/FUSION52260.2023.10224196

Dhaka, A. K., Catalina, A., Andersen, M. R., Magnusson, M. n., Huggins, J., & Vehtari, A. (2020). Robust, Accurate Stochastic Optimization for Variational Inference. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, & H. Lin (Eds.), *Advances in Neural Information Processing Systems* (Vol. 33, p. 10961-10973). Curran Associates, Inc. Retrieved from `https://proceedings.neurips.cc/paper/2020/file/7cac11e2f46ed46c339ec3d569853759-Paper.pdf`

Dinh, L., Krueger, D., & Bengio, Y. (2015). NICE: Non-linear Independent Components Estimation. In Y. Bengio & Y. LeCun (Eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings.* Retrieved from `http://arxiv.org/abs/1410.8516`

Dinh, L., Sohl-Dickstein, J., & Bengio, S. (2017). Density estimation using Real NVP. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. Retrieved from https://openreview.net/forum?id=HkpbnH9lx

Ditlevsen, S., & Samson, A. (2013). Introduction to Stochastic Models in Biology. In M. Bachar, J. Batzel, & S. Ditlevsen (Eds.), *Stochastic Biomathematical Models: with Applications to Neuronal Modeling* (pp. 3–35). Berlin, Heidelberg: Springer. Retrieved 2025-04-03, from https://doi.org/10.1007/978-3-642-32157-3_1 doi: 10.1007/978-3-642-32157-3_1

Duane, S., Kennedy, A., Pendleton, B. J., & Roweth, D. (1987). Hybrid Monte Carlo. *Physics Letters B*, *195*(2), 216-222. doi: https://doi.org/10.1016/0370-2693(87)91197-X

Evensen, G. (1994). Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research: Oceans*, *99*(C5), 10143–10162. Retrieved 2025-04-29, from https://onlinelibrary.wiley.com/doi/abs/10.1029/94JC00572 (_eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/94JC00572) doi: 10.1029/94JC00572

Fuchs, C. (2013). *Inference for Diffusion Processes: With Applications in Life Sciences*. Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from https://doi.org/10.1007/978-3-642-25969-2 doi: 10.1007/978-3-642-25969-2

Gelman, A., Goodrich, B., Gabry, J., & Vehtari, A. (2019). R-squared for Bayesian Regression Models. *The American Statistician*, *73*(3). Retrieved from https://doi.org/10.1080/00031305.2018.1549100 doi: 10.1080/00031305.2018.1549100

Gelman, A., Hwang, J., & Vehtari, A. (2014). Understanding predictive information criteria for Bayesian models. *Statistics and Computing*, *24*(6). doi: 10.1007/s11222-013-9416-2

Geman, S., & Geman, D. (1987). Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. In M. A. Fischler & O. Firschein (Eds.), *Readings in Computer Vision*. San Francisco (CA): Morgan Kaufmann. Retrieved from https://www.sciencedirect.com/science/article/pii/B978008051581650057X doi: 10.1016/B978-0-08-051581-6.50057-X

Georgiou, K., Malhotra, A., Wieder, W. R., Ennis, J. H., Hartman, M. D., Sulman, B. N., ... Jackson, R. B. (2021). Divergent controls of soil organic carbon between observations and process-based models. *Biogeochemistry*. Retrieved from https://doi.org/10.1007/s10533-021-00819-2 doi: 10.1007/s10533-021-00819-2

Ghosh, A., Honoré, A., & Chatterjee, S. (2023). Danse: Data-driven non-linear state estimation of model-free process in unsupervised bayesian setup. In *2023 31st european signal processing conference (eusipco)* (p. 870-874). doi: 10.23919/EUSIPCO58844.2023.10289946

Giordano, R., Broderick, T., & Jordan, M. I. (2018). Covariances, Robustness, and Variational Bayes. *Journal of Machine Learning Research*, *19*(51). Retrieved from http://jmlr.org/papers/v19/17-670.html

Giweta, M. (2020). Role of litter production and its decomposition, and factors affecting the processes in a tropical forest ecosystem: a review. *Journal of Ecology and Environment*, *44*(1). Retrieved from https://doi.org/10.1186/s41610-020-0151-2 doi: 10.1186/s41610-020-0151-2

Golightly, A., & Kypraios, T. (2018). Efficient SMC$^2$ schemes for stochastic kinetic models. *Statistics and Computing*, *28*(6). Retrieved from https://doi.org/10.1007/s11222-017-9789-8 doi: 10.1007/s11222-017-9789-8

Golightly, A., & Wilkinson, D. (2010). Markov chain Monte Carlo algorithms for SDE parameter estimation. In N. D. Lawrence, M. Girolami, M. Rattray, &

G. Sanguinetti (Eds.), . Cambridge, MA, USA: MIT Press.

Golightly, A., & Wilkinson, D. J. (2005). Bayesian Inference for Stochastic Kinetic Models Using a Diffusion Approximation. *Biometrics*, *61*(3), 781-788. Retrieved from https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1541-0420.2005.00345.x doi: https://doi.org/10.1111/j.1541-0420.2005.00345.x

Golightly, A., & Wilkinson, D. J. (2006). Bayesian sequential inference for nonlinear multivariate diffusions. *Statistics and Computing*, *16*(4). Retrieved from https://doi.org/10.1007/s11222-006-9392-x doi: 10.1007/s11222-006-9392-x

Golightly, A., & Wilkinson, D. J. (2011). Bayesian parameter inference for stochastic biochemical network models using particle Markov chain Monte Carlo. *Interface Focus*, *1*(6). Retrieved from https://pubmed.ncbi.nlm.nih.gov/23226583 doi: 10.1098/rsfs.2011.0047

Goyal, P., Dollár, P., Girshick, R., Noordhuis, P., Wesolowski, L., Kyrola, A., ... He, K. (2017). *Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour.* arXiv. Retrieved from https://arxiv.org/abs/1706.02677 doi: 10.48550/ARXIV.1706.02677

Hararuk, O., & Luo, Y. (2014). Improvement of global litter turnover rate predictions using a Bayesian MCMC approach. *Ecosphere*, *5*(12). Retrieved from https://esajournals.onlinelibrary.wiley.com/doi/abs/10.1890/ES14-00092.1 doi: 10.1890/ES14-00092.1

Hararuk, O., Xia, J., & Luo, Y. (2014). Evaluation and improvement of a global land model against soil carbon data using a Bayesian Markov chain Monte Carlo method. *Journal of Geophysical Research: Biogeosciences*, *119*(3). Retrieved from https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1002/2013JG002535 doi: 10.1002/2013JG002535

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., ... Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825). Retrieved from https://doi.org/10.1038/s41586-020-2649-2 doi: 10.1038/s41586-020-2649-2

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. doi: 10.1109/CVPR.2016.90

Hoffman, M. D., Blei, D. M., Wang, C., & Paisley, J. (2013). Stochastic Variational Inference. *Journal of Machine Learning Research*, *14*(40). Retrieved from http://jmlr.org/papers/v14/hoffman13a.html

Hoffman, M. D., & Gelman, A. (2014). The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, *15*(47). Retrieved from http://jmlr.org/papers/v15/hoffman14a.html

Ioffe, S. (2017). Batch Renormalization: Towards Reducing Minibatch Dependence in Batch-Normalized Models. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc.

Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. JMLR.org.

Januszewski, M., & Kostur, M. (2010, January). Accelerating numerical solution of stochastic differential equations with CUDA. *Computer Physics Communications*, *181*(1), 183–188. Retrieved 2025-04-10, from https://www.sciencedirect.com/science/article/pii/S0010465509002999 doi: 10.1016/j.cpc.2009.09.009

Jia, J., & Benson, A. R. (2019). Neural Jump Stochastic Differential Equations. In *Proceedings of the 33rd International Conference on Neural Information*

*Processing Systems.* Red Hook, NY, USA: Curran Associates Inc.

Joyce, J. M. (2011). Kullback-Leibler Divergence. In M. Lovric (Ed.), *International Encyclopedia of Statistical Science.* Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from https://doi.org/10.1007/978-3-642-04898-2_327 doi: 10.1007/978-3-642-04898-2_327

Kalman, R. E. (1960). A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, *82*(1). Retrieved from https://doi.org/10.1115/1.3662552 doi: 10.1115/1.3662552

Katzfuss, M., Stroud, J. R., & Wikle, C. K. (2016). Understanding the Ensemble Kalman Filter. *The American Statistician*, *70*(4), 350–357. doi: 10.1080/00031305.2016.1141709

Keller, J., Hendricks Franssen, H.-J., & Marquart, G. (2018). Comparing Seven Variants of the Ensemble Kalman Filter: How Many Synthetic Experiments Are Needed? *Water Resources Research*, *54*(9), 6299–6318. Retrieved 2025-04-29, from https://onlinelibrary.wiley.com/doi/abs/10.1029/2018WR023374 (_eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/2018WR023374) doi: 10.1029/2018WR023374

Kingma, D. P., & Ba, J. (2015). Adam: A Method for Stochastic Optimization. In Y. Bengio & Y. LeCun (Eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.* Retrieved from http://arxiv.org/abs/1412.6980

Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., & Welling, M. (2016). Improved Variational Inference with Inverse Autoregressive Flow. In *Proceedings of the 30th International Conference on Neural Information Processing Systems.* Red Hook, NY, USA: Curran Associates Inc.

Kingma, D. P., & Welling, M. (2014). Auto-Encoding Variational Bayes. In Y. Bengio & Y. LeCun (Eds.), *2nd international conference on learning representations, ICLR 2014, banff, ab, canada, april 14-16, 2014, conference track proceedings.* Retrieved from http://arxiv.org/abs/1312.6114

Kobyzev, I., Prince, S., & Brubaker, M. (2020). Normalizing Flows: An Introduction and Review of Current Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Retrieved from http://dx.doi.org/10.1109/TPAMI.2020.2992934 doi: 10.1109/tpami.2020.2992934

Köhler, J., Krämer, A., & Noe, F. (2021). Smooth Normalizing Flows. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, & J. W. Vaughan (Eds.), *Advances in neural information processing systems* (Vol. 34, pp. 2796–2809). Curran Associates, Inc.

Kong, Z., & Chaudhuri, K. (2020, 26–28 Aug). The Expressive Power of a Class of Normalizing Flow Models. In S. Chiappa & R. Calandra (Eds.), *Proceedings of the twenty third international conference on artificial intelligence and statistics* (Vol. 108, pp. 3599–3609). PMLR.

Krishnan, R. G., Shalit, U., & Sontag, D. (2017). Structured Inference Networks for Nonlinear State Space Models. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence.* AAAI Press.

Kucukelbir, A., Tran, D., Ranganath, R., Gelman, A., & Blei, D. M. (2017). Automatic Differentiation Variational Inference. *Journal of Machine Learning Research*, *18*(1), 430-474.

Kullback, S., & Leibler, R. A. (1951). On Information and Sufficiency. *The Annals of Mathematical Statistics*, *22*(1). Retrieved from https://doi.org/10.1214/aoms/1177729694 doi: 10.1214/aoms/1177729694

Lambert, B., Lei, C. L., Robinson, M., Clerx, M., Creswell, R., Ghosh, S., ... Gavaghan, D. J. (2023). Autocorrelated measurement processes and inference for ordinary differential equation models of biological systems. *Journal of The Royal Society Interface*, *20*(199), 20220725. Retrieved from

https://royalsocietypublishing.org/doi/abs/10.1098/rsif.2022.0725 doi: 10.1098/rsif.2022.0725

Lee, H., Pabbaraju, C., Sevekari, A., & Risteski, A. (2021). *Universal Approximation for Log-concave Distributions using Well-conditioned Normalizing Flows.* arXiv. Retrieved from https://arxiv.org/abs/2107.02951 doi: 10.48550/ARXIV.2107.02951

Li, J., Wang, G., Allison, S. D., Mayes, M. A., & Luo, Y. (2014). Soil carbon sensitivity to temperature and carbon use efficiency compared across microbial-ecosystem models of varying complexity. *Biogeochemistry*, *119*(1-3). Retrieved from https://doi.org/10.1007/s10533-013-9948-8 doi: 10.1007/s10533-013-9948-8

Li, J., Wang, G., Mayes, M. A., Allison, S. D., Frey, S. D., Shi, Z., ... Melillo, J. M. (2019). Reduced carbon use efficiency and increased microbial turnover with soil warming. *Global Change Biology*, *25*(3). Retrieved from https://onlinelibrary.wiley.com/doi/abs/10.1111/gcb.14517 doi: 10.1111/gcb.14517

Li, T., Sun, S., Sattar, T. P., & Corchado, J. M. (2014). Fight sample degeneracy and impoverishment in particle filters: A review of intelligent approaches. *Expert Systems with Applications*, *41*(8), 3944-3954. doi: https://doi.org/10.1016/j.eswa.2013.12.031

Li, X., Wong, T.-K. L., Chen, R. T. Q., & Duvenaud, D. (2020). Scalable Gradients for Stochastic Differential Equations. In S. Chiappa & R. Calandra (Eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics* (Vol. 108). PMLR. Retrieved from https://proceedings.mlr.press/v108/li20i.html

Lu, D., Ricciuto, D., Walker, A., Safta, C., & Munger, W. (2017, September). Bayesian calibration of terrestrial ecosystem models: a study of advanced Markov chain Monte Carlo methods. *Biogeosciences*, *14*(18), 4295–4314. Retrieved 2025-04-03, from https://bg.copernicus.org/articles/14/4295/2017/ (Publisher: Copernicus GmbH) doi: 10.5194/bg-14-4295-2017

Luo, Y., Ahlström, A., Allison, S. D., Batjes, N. H., Brovkin, V., Carvalhais, N., ... Zhou, T. (2016). Toward more realistic projections of soil carbon dynamics by Earth system models. *Global Biogeochemical Cycles*, *30*(1). Retrieved from https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1002/2015GB005239 doi: 10.1002/2015GB005239

Manzoni, S., & Porporato, A. (2009). Soil carbon and nitrogen mineralization: Theory and models across scales. *Soil Biology and Biochemistry*, *41*(7). Retrieved from http://dx.doi.org/10.1016/j.soilbio.2009.02.031 doi: 10.1016/j.soilbio.2009.02.031

Maruyama, G. (1955). Continuous Markov processes and stochastic equations. *Rendiconti del Circolo Matematico di Palermo*, *4*(1). Retrieved from https://doi.org/10.1007/BF02846028 doi: 10.1007/BF02846028

Melillo, J. M., Frey, S. D., DeAngelis, K. M., Werner, W. J., Bernard, M. J., Bowles, F. P., ... Grandy, A. S. (2017). Long-term pattern and magnitude of soil carbon feedback to the climate system in a warming world. *Science*, *358*(6359). doi: 10.1126/science.aan2874

Nemeth, C., & Fearnhead, P. (2021). Stochastic Gradient Markov Chain Monte Carlo. *Journal of the American Statistical Association*, *116*(533). Retrieved from https://doi.org/10.1080/01621459.2020.1847120 doi: 10.1080/01621459.2020.1847120

O'Neill, B. C., Kriegler, E., Ebi, K. L., Kemp-Benedict, E., Riahi, K., Rothman, D. S., ... Solecki, W. (2017). The roads ahead: Narratives for shared socioeconomic pathways describing world futures in the 21st century. *Global Environmental Change*, *42*. Retrieved from https://www.sciencedirect.com/science/article/pii/S0959378015000060 doi:

10.1016/j.gloenvcha.2015.01.004

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., & Lakshminarayanan, B. (2021). Normalizing Flows for Probabilistic Modeling and Inference. *Journal of Machine Learning Research*, *22*(57). Retrieved from http://jmlr.org/papers/v22/19-1028.html

Papamakarios, G., Pavlakou, T., & Murray, I. (2017). Masked Autoregressive Flow for Density Estimation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems.* Red Hook, NY, USA: Curran Associates Inc.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 32.* Curran Associates, Inc. Retrieved from http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Perez-Cruz, F. (2008). Kullback-Leibler divergence estimation of continuous distributions. In *2008 IEEE International Symposium on Information Theory.* doi: 10.1109/ISIT.2008.4595271

Plummer, M. (2003). JAGS: A Program for Analysis of Bayesian Graphical Models using Gibbs Sampling. In K. Hornik, F. Leisch, & A. Zeileis (Eds.), *Proceedings of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003)* (Vol. 3). Vienna, Austria.

Raczka, B., Hoar, T. J., Duarte, H. F., Fox, A. M., Anderson, J. L., Bowling, D. R., & Lin, J. C. (2021). Improving CLM5.0 Biomass and Carbon Exchange Across the Western United States Using a Data Assimilation System. *Journal of Advances in Modeling Earth Systems*, *13*(7). Retrieved from https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2020MS002421 (e2020MS002421 2020MS002421) doi: 10.1029/2020MS002421

Rauch, H. E., Tung, F., & Striebel, C. T. (1965). Maximum likelihood estimates of linear dynamic systems. *AIAA journal*, *3*(8).

Reid, N. M. (2015). Approximate likelihoods. In L. Guo & Z.-M. Ma (Eds.), *Proceedings of the 8th International Congress on Industrial and Applied Mathematics.* Beijing, China: Higher Education Press.

Ruiz, F. R., Titsias RC AUEB, M., & Blei, D. (2016). The Generalized Reparameterization Gradient. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems* (Vol. 29). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper/2016/file/f718499c1c8cef6730f9fd03c8125cab-Paper.pdf

Ryder, T., Golightly, A., McGough, A. S., & Prangle, D. (2018). Black-Box Variational Inference for Stochastic Differential Equations. In J. Dy & A. Krause (Eds.), *Proceedings of the 35th International Conference on Machine Learning* (Vol. 80). PMLR. Retrieved from https://proceedings.mlr.press/v80/ryder18a.html

Ryder, T., Prangle, D., Golightly, A., & Matthews, I. (2021). The neural moving average model for scalable variational inference of state space models. In C. de Campos & M. H. Maathuis (Eds.), *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence* (Vol. 161, pp. 12–22). PMLR. Retrieved from https://proceedings.mlr.press/v161/ryder21a.html

Saifuddin, M., Abramoff, R. Z., Davidson, E. A., Dietze, M. C., & Finzi, A. C. (2021). Identifying Data Needed to Reduce Parameter Uncertainty in a Coupled Microbial Soil C and N Decomposition Model. *Journal of Geophysical Research: Biogeosciences*, *126*(12). Retrieved from https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2021JG006593 doi: 10.1029/2021JG006593

Salimans, T., Kingma, D., & Welling, M. (2015). Markov Chain Monte Carlo and Variational Inference: Bridging the Gap. In F. Bach & D. Blei (Eds.), *Proceedings of the 32nd International Conference on Machine Learning* (Vol. 37). Lille, France: PMLR. Retrieved from https://proceedings.mlr.press/v37/salimans15.html

Salimans, T., & Knowles, D. A. (2013). Fixed-Form Variational Posterior Approximation through Stochastic Linear Regression. *Bayesian Analysis*, *8*(4). Retrieved from https://doi.org/10.1214/13-BA858 doi: 10.1214/13-BA858

Salvatier, J., Wiecki, T. V., & Fonnesbeck, C. (2016). Probabilistic programming in Python using PyMC3. *PeerJ Computer Science*, *2*. Retrieved from https://doi.org/10.7717/peerj-cs.55 doi: 10.7717/peerj-cs.55

Särkkä, S. (2013). *Bayesian Filtering and Smoothing*. Cambridge: Cambridge University Press. Retrieved from https://www.cambridge.org/core/books/bayesian-filtering-and-smoothing/C372FB31C5D9A100F8476C1B23721A67 doi: 10.1017/CBO9781139344203

Särkkä, S., & Solin, A. (2019). *Applied Stochastic Differential Equations*. Cambridge University Press. doi: 10.1017/9781108186735

Shen, Z., & Tang, Y. (2015). A modified ensemble Kalman particle filter for non-gaussian systems with nonlinear measurement functions. *Journal of Advances in Modeling Earth Systems*, *7*(1), 50-66. doi: https://doi.org/10.1002/2014MS000373

Sujono, D., Xie, H. W., Allison, S., & Sudderth, E. B. (2022). Variational Inference for Soil Biogeochemical Models. In *ICML 2022 2nd AI for Science Workshop*. Retrieved from https://openreview.net/forum?id=3_HrwqrPd4U

Sulman, B. N., Moore, J. A., Abramoff, R., Averill, C., Kivlin, S., Georgiou, K., ... Classen, A. T. (2018). Multiple models and experiments underscore large uncertainty in soil carbon dynamics. *Biogeochemistry*, *141*(2). doi: 10.1007/s10533-018-0509-z

Sulman, B. N., Phillips, R. P., Oishi, A. C., Shevliakova, E., & Pacala, S. W. (2014). Microbe-driven turnover offsets mineral-mediated storage of soil carbon under elevated $CO_2$. *Nature Climate Change*, *4*(12). doi: 10.1038/nclimate2436

Summers, C., & Dinneen, M. J. (2020). Four Things Everyone Should Know to Improve Batch Normalization. In *International Conference on Learning Representations*. Retrieved from https://openreview.net/forum?id=HJx8HANFDH

Todd-Brown, K. E. O., Randerson, J. T., Hopkins, F., Arora, V., Hajima, T., Jones, C., ... Allison, S. D. (2014). Changes in soil organic carbon storage predicted by earth system models during the 21st century. *Biogeosciences*, *11*(8), 2341-2356. Retrieved from https://bg.copernicus.org/articles/11/2341/2014/ doi: 10.5194/bg-11-2341-2014

Todd-Brown, K. E. O., Randerson, J. T., Post, W. M., Hoffman, F. M., Tarnocai, C., Schuur, E. A. G., & Allison, S. D. (2013). Causes of variation in soil carbon simulations from CMIP5 Earth system models and comparison with observations. *Biogeosciences*, *10*(3). Retrieved from https://bg.copernicus.org/articles/10/1717/2013/ doi: 10.5194/bg-10-1717-2013

Tzen, B., & Raginsky, M. (2019). *Neural Stochastic Differential Equations: Deep Latent Gaussian Models in the Diffusion Limit*. arXiv. Retrieved from https://arxiv.org/abs/1905.09883 doi: 10.48550/ARXIV.1905.09883

Vehtari, A., Gelman, A., & Gabry, J. (2017). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *Statistics and Computing*, *27*(5). doi: 10.1007/s11222-016-9696-4

Wang, S., Luo, Y., & Niu, S. (2022). Reparameterization Required After Model Structure Changes From Carbon Only to Carbon-Nitrogen Coupling. *Journal of Advances in Modeling Earth Systems*, *14*(4). Retrieved from https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2021MS002798 doi: 10.1029/2021MS002798

Wang, X., Guan, X., & Guo, L. (2025). *Flow-based Bayesian filtering for high-dimensional nonlinear stochastic dynamical systems.* arXiv. Retrieved 2025-04-29, from https://arxiv.org/html/2502.16232v2#bib.bib44

Whitaker, G. A. (2016). *Bayesian inference for stochastic differential mixed-effects models* (Doctoral dissertation, Newcastle University). Retrieved from http://theses.ncl.ac.uk/jspui/handle/10443/3398

Whitaker, G. A., Golightly, A., Boys, R. J., & Sherlock, C. (2017). Bayesian Inference for Diffusion-Driven Mixed-Effects Models. *Bayesian Analysis*, *12*(2). Retrieved from https://doi.org/10.1214/16-BA1009 doi: 10.1214/16-BA1009

Wieder, W. R., Boehnert, J., & Bonan, G. B. (2014). Evaluating soil biogeochemistry parameterizations in Earth system models with observations. *Global Biogeochemical Cycles*, *28*(3). Retrieved from https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1002/2013GB004665 doi: 10.1002/2013GB004665

Wieder, W. R., Grandy, A. S., Kallenbach, C. M., Taylor, P. G., & Bonan, G. B. (2015). Representing life in the Earth system with soil microbial functional traits in the MIMICS model. *Geoscientific Model Development*, *8*(6). doi: 10.5194/gmd-8-1789-2015

Wiqvist, S., Golightly, A., McLean, A. T., & Picchini, U. (2021). Efficient inference for stochastic differential equation mixed-effects models using correlated particle pseudo-marginal algorithms. *Computational Statistics & Data Analysis*, *157*. Retrieved from https://www.sciencedirect.com/science/article/pii/S0167947320302425 doi: 10.1016/j.csda.2020.107151

Wood, T. E., González, G., Silver, W. L., Reed, S. C., & Cavaleri, M. A. (2019). On the shoulders of giants: Continuing the legacy of large-scale ecosystem manipulation experiments in Puerto Rico. *Forests*, *10*(3). doi: 10.3390/f10030210

Xie, H. W. (2024). *Variational inference soil biogeochemical model experiment scripts and synthetic data [dataset] [software].* Retrieved from https://doi.org/10.5281/zenodo.13917102

Xie, H. W., Romero-Olivares, A. L., Guindani, M., & Allison, S. D. (2020). A Bayesian approach to evaluation of soil biogeochemical models. *Biogeosciences*, *17*(15). Retrieved from https://bg.copernicus.org/articles/17/4043/2020/ doi: 10.5194/bg-17-4043-2020

Yao, Y., Vehtari, A., Simpson, D., & Gelman, A. (2018). Yes, but Did It Work?: Evaluating Variational Inference. In J. Dy & A. Krause (Eds.), *Proceedings of the 35th International Conference on Machine Learning* (Vol. 80). PMLR. Retrieved from https://proceedings.mlr.press/v80/yao18a.html

You, K., Long, M., Wang, J., & Jordan, M. I. (2019). *How Does Learning Rate Decay Help Modern Neural Networks?* arXiv. Retrieved from https://arxiv.org/abs/1908.01878 doi: 10.48550/ARXIV.1908.01878

Zhai, S., Zhang, R., Nakkiran, P., Berthelot, D., Gu, J., Zheng, H., ... Susskind, J. (2025). *Normalizing Flows are Capable Generative Models.* Retrieved from https://arxiv.org/abs/2412.06329

Zhu, Q., Bi, W., Liu, X., Ma, X., Li, X., & Wu, D. (2020). A Batch Normalized Inference Network Keeps the KL Vanishing Away. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.* Online: Association for Computational Linguistics. Retrieved from https://aclanthology.org/2020.acl-main.235 doi: 10.18653/v1/2020.acl-main.235