Composing the Score for Stable and Simulation-Efficient Amortized Inference at Scale

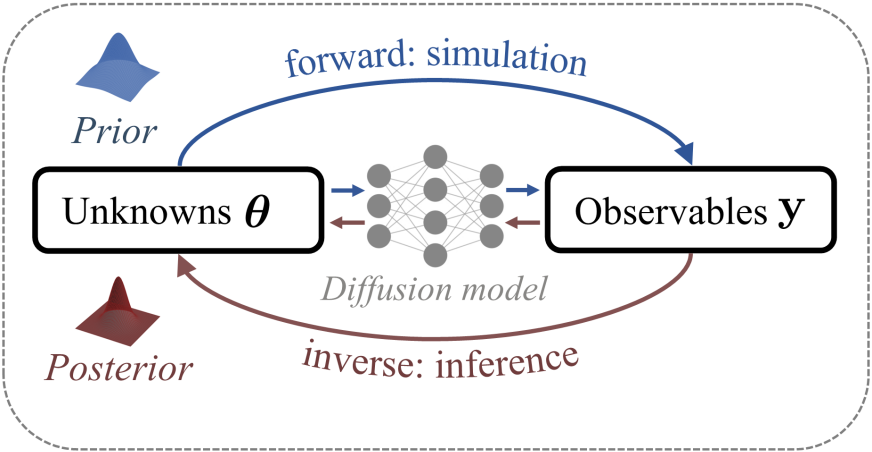
Stefan T. Radev and Indispensible Others

Center for Modeling, Simulation, & Imaging in Medicine (CeMSIM)
Department of Cognitive Science
Rensselaer Polytechnic Institute

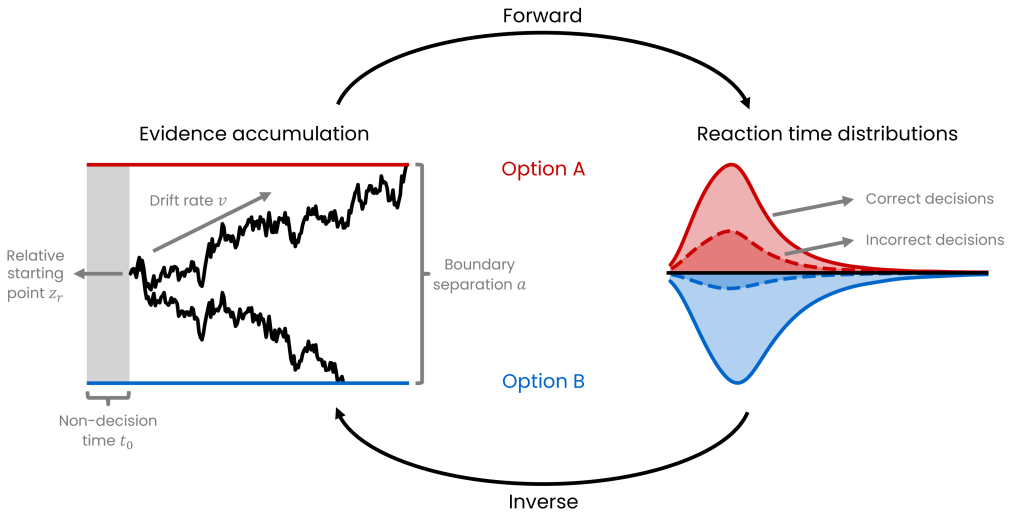
October 4, 2025

Agenda

Problem Setting



Example: Decision-Making



Simulators vs. Likelihoods

Likelihood-Based (Explicit)

Bayesian model $p(\boldsymbol{\theta}, \mathbf{Y})$:

$$\boldsymbol{\theta} \sim p(\boldsymbol{\theta})$$
$$\mathbf{Y} \sim p(\mathbf{Y} \mid \boldsymbol{\theta})$$

- Prior $p(\boldsymbol{\theta})$ can be sampled and evaluated.
- Data model $p(\mathbf{Y} \mid \boldsymbol{\theta})$ can be sampled and evaluated.

Simulation-Based (Implicit)

The same Bayesian model $p(\boldsymbol{\theta}, \mathbf{Y})$:

$$\boldsymbol{\theta} \sim p(\boldsymbol{\theta})$$
$$\mathbf{Y} = \mathrm{Sim}(\boldsymbol{\theta}, \mathbf{Z}), \ \mathbf{Z} \sim \mathrm{RNG}(\cdot)$$

- Prior $p(\boldsymbol{\theta})$ can be sampled and optionally evaluated.
- Data model $p(\mathbf{Y} \mid \boldsymbol{\theta})$ can be sampled but not evaluated.

The Hardships of Embracing Uncertainty

- Bayesian inference is notationally straightforward:

$$p(\boldsymbol{\theta} \mid \mathbf{Y}) \propto p(\mathbf{Y} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta}) \tag{1}$$

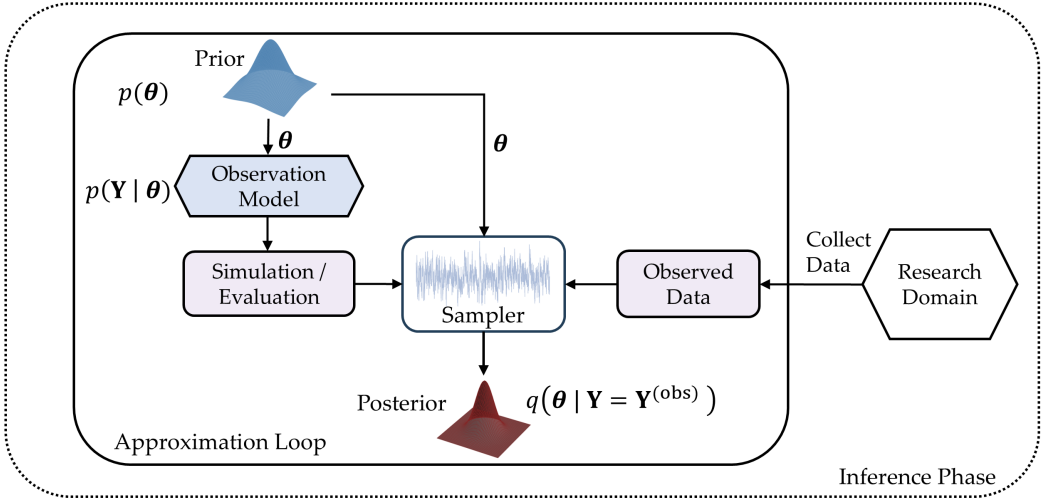
- Bayesian inference is hard because integration is hard:

$$p(\boldsymbol{\theta} \mid \mathbf{Y}) = p(\mathbf{Y} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta}) \times \left(\int_{\Theta} p(\mathbf{Y} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta} \right)^{-1} \tag{2}$$

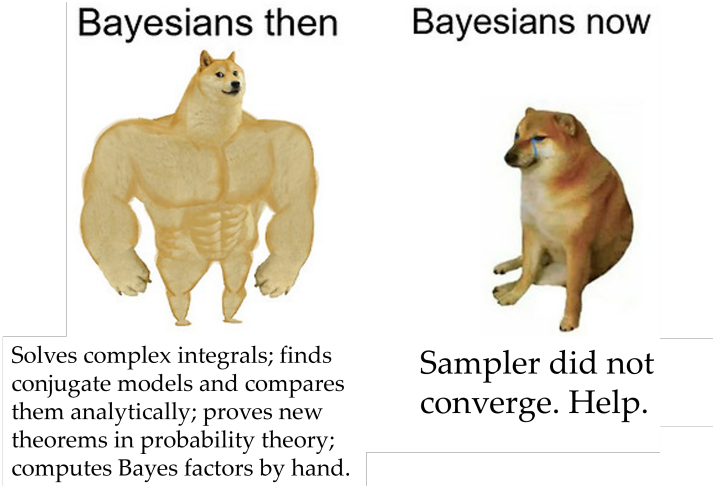
- Simulation-based inference is harder because integration can be twice as hard:

$$p(\boldsymbol{\theta} \mid \mathbf{Y}) = \int_{\mathcal{Z}} p(\mathbf{Y}, \mathbf{Z} \mid \boldsymbol{\theta}) \mathrm{d}\mathbf{Z} \times p(\boldsymbol{\theta}) \times \left(\int_{\Theta} p(\mathbf{Y} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta} \right)^{-1} \tag{3}$$

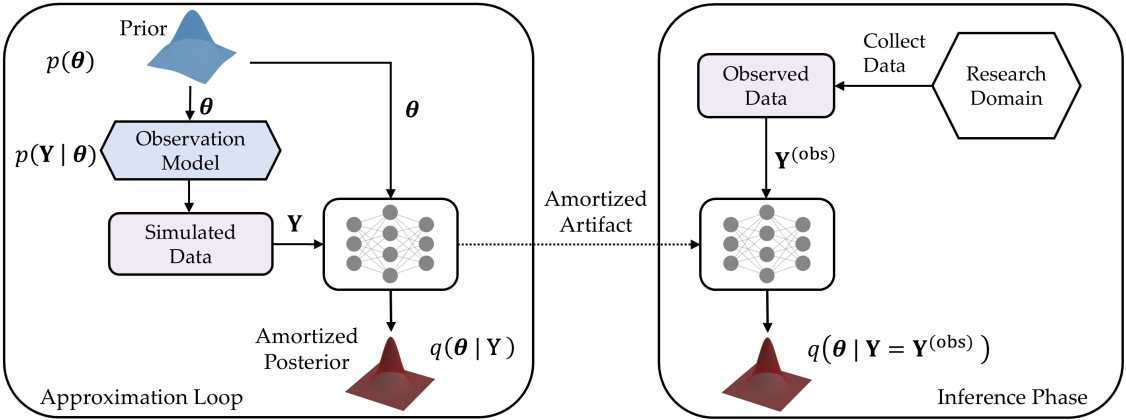
Non-Amortized Bayesian Inference



Bayesians Now and Then



Amortized Bayesian Inference



Bayesians Now

Bayesians Now

Sampler did not converge. Help.

Also Bayesians Now

Network did not converge. Help.

Objective

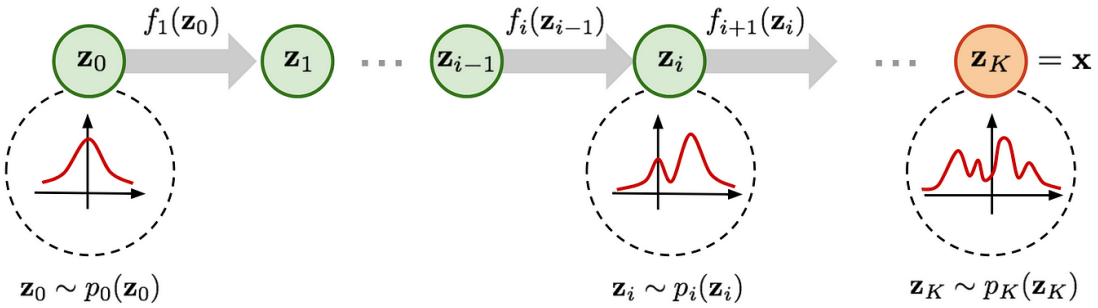
- In regular amortized inference, we minimize a strictly proper scoring loss $\mathcal{S}$ in expectation over the Bayesian model $p(\boldsymbol{\theta}, \mathbf{Y})$:

$$\min_q \left\{ \mathbb{E}_{p(\boldsymbol{\theta},\mathbf{Y})} \Big[\mathcal{S}(q(\cdot \mid \mathbf{Y}), \boldsymbol{\theta}) \Big] \approx \frac{1}{B} \sum_{b=1}^{B} \mathcal{S}(q(\cdot \mid \mathbf{Y}^{(b)}), \boldsymbol{\theta}^{(b)}) \right\}. \tag{4}$$

- If we want to extend the amortization scope, we can generalize over any number of context variables $\mathbf{C}$ by adding additional conditions:

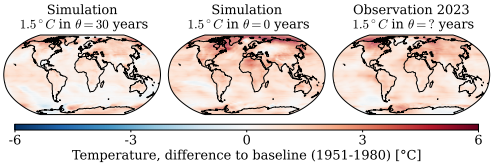
$$\min_q \mathbb{E}_{p(\boldsymbol{\theta},\mathbf{Y},\mathbf{C})} \Big[\mathcal{S}(q(\cdot \mid \mathbf{Y}, \mathbf{C}), \boldsymbol{\theta}) \Big] \tag{5}$$

Generative Networks: Normalizing Flows

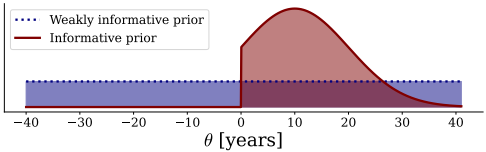


Key idea: during training, learn to transform complex samples (e.g., from the posterior) into simple ones (e.g., Gaussian); during inference, sample from the simple distribution and transform back to the complex distribution. Image source: Lil'Log.

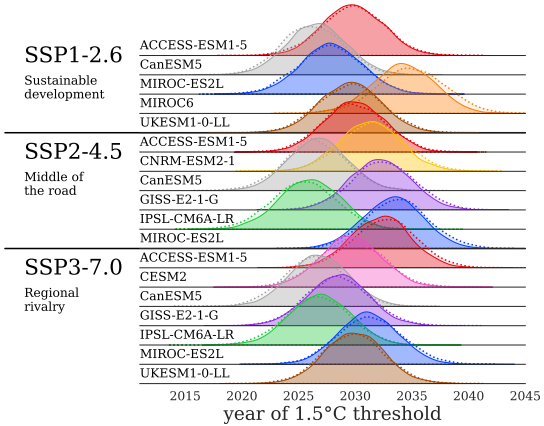
Sensitivity-Aware Inference of Critical Thresholds



Simulation snapshots (**left**, **center**) and the empirical observation (**right**).

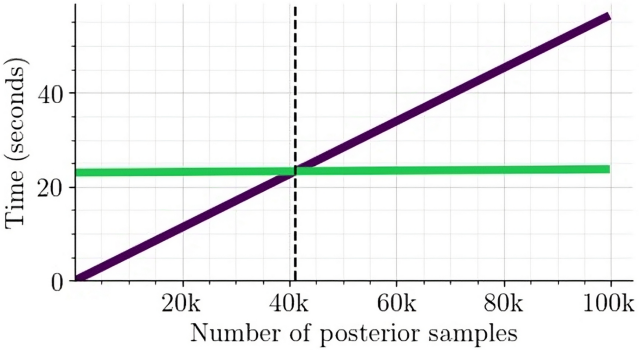


Qualitatively different prior distributions.



Figures from Elsemüller et al. (2024).

Breakeven Points



MCMC:

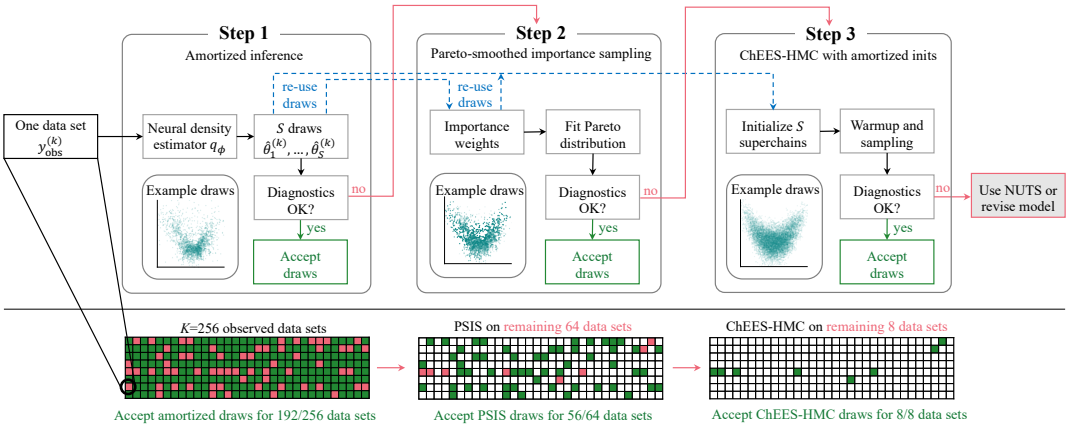
- no upfront training
- 5.66 seconds for 10k posterior draws

Amortized Inference:

- 23 seconds upfront training
- 0.07 seconds for 10k posterior draws

Complementarity: Towards an Amortized Workflow

- Towards an automated amortized Bayesian workflow (Schmitt et al., 2024):



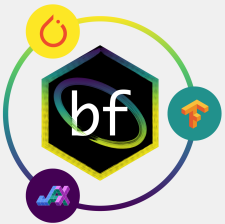
Agenda

Overview of BayesFlow

❶ SIMULATE.

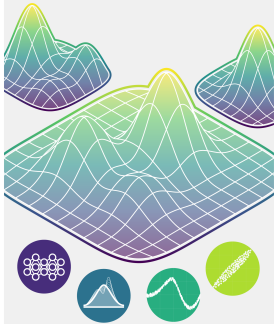
Generate data from
any simulation you like

❷ AMORTIZE.



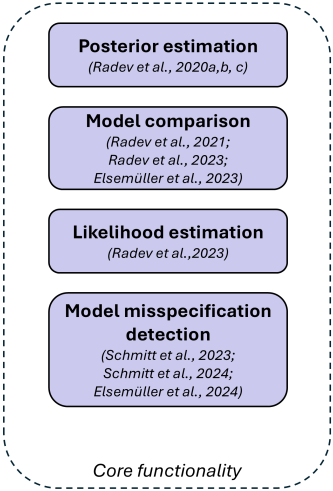
Part of the Keras ecosystem
with **multi-backend support**

❸ LEARN!

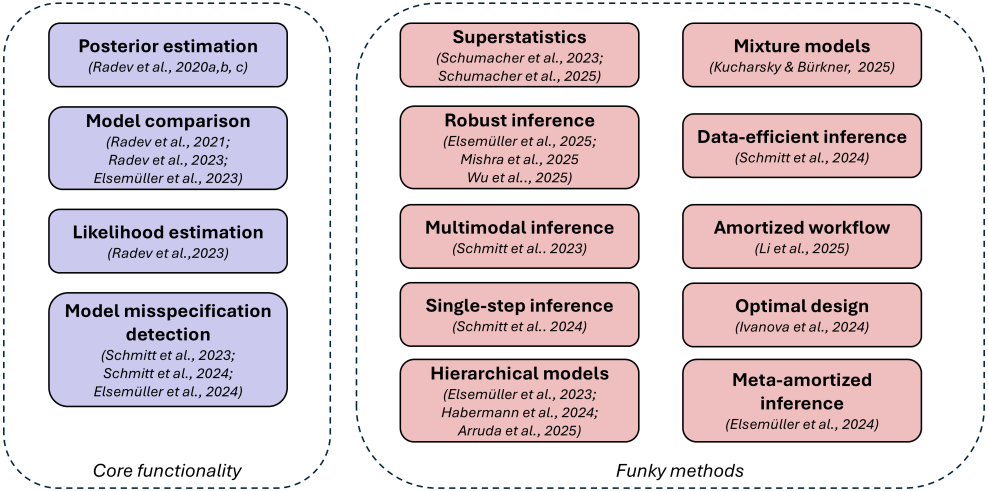


Super-powerful **generative AI**
with **robust diagnostics**

Functions and Opportunities



Functions and Opportunities



Available Generative Model Families

Generative Family	**Architecture**	**Sampling**	**Density Evaluation**
Normalizing Flows	Constrained	Single-step	Fast
Free-Form Flows	Semi-Constrained	Single-step	Moderate
Diffusion Models	Unconstrained	Multi-step	Slow
Flow Matching	Unconstrained	Multi-step	Slow
Consistency Models	Unconstrained	Few-step	N/A

Join Us

- GitHub page: https://github.com/bayesflow-org/bayesflow
- Awesome amortized list: https://github.com/bayesflow-org/awesome-amortized-inference
- Project page: https://bayesflow.org/
- Forums: discuss.bayesflow.org
- BlueSky: https://bsky.app/profile/bayesflow.org
- Zulip: https://bayesflow.zulipchat.com/

Agenda

Credits

Compositional amortized inference for large-scale hierarchical Bayesian models

Jonas Arruda
Bonn Center for Mathematical Life Sciences,
& Life and Medical Sciences Institute
University of Bonn, Germany

Vikas Pandey
Center for Modeling, Simulation,
& Imaging in Medicine (CeMSIM)
Rensselaer Polytechnic Institute, NY, USA

Catherine Sherry
Molecular and Cellular Physiology
Albany Medical College, NY, USA

Margarida Barroso
Molecular and Cellular Physiology
Albany Medical College, NY, USA

Xavier Intes
Center for Modeling, Simulation,
& Imaging in Medicine (CeMSIM)
Rensselaer Polytechnic Institute, NY, USA

Jan Hasenauer
Bonn Center for Mathematical Life Sciences,
& Life and Medical Sciences Institute
University of Bonn, Germany

Stefan T. Radev
Center for Modeling, Simulation,
& Imaging in Medicine (CeMSIM)
Rensselaer Polytechnic Institute, NY, USA

Abstract

Amortized Bayesian inference (ABI) has emerged as a powerful simulation-based approach for estimating complex mechanistic models, offering fast posterior sampling via generative neural networks. However, extending ABI to hierarchical models, a cornerstone of modern Bayesian analysis, remains a major challenge due to the need to simulate massive data sets and estimate thousands of parameters. In this work, we build on compositional score matching (CSM), a divide-and-conquer strategy for Bayesian updating using diffusion models. To address existing stability issues of CSM in dealing with large data sets, we couple adaptive solvers with a novel, error-damping compositional estimator. Our estimator remains stable even with hundreds of thousands of data points and parameters. We validate our approach on a controlled toy example, a high-dimensional autoregressive model, and a real-world advanced microscopy application involving over 750,000 parameters.

Jonas Arruda
(University of Bonn)

Vikas Pandey
(RPI)

The Problem

- Many datasets contain complex dependency structures caused by natural groupings (e.g., experiments) or repeated noisy measurements of the same units (e.g., events).
- In modern Bayesian data analysis, hierarchical Bayesian models are the standard approach to tackle these dependencies (Gelman et al., 2013; McElreath, 2018).
- From a Bayesian perspective, any parametric data model $p(\mathbf{Y} \mid \boldsymbol{\theta})$ can incorporate multilevel structure via a hierarchical prior. Example two-level model:

$$\mathbf{Y}_j \sim p(\mathbf{Y}_j \mid \boldsymbol{\theta}_j, \boldsymbol{\eta}), \quad \boldsymbol{\theta}_j \sim p(\boldsymbol{\theta} \mid \boldsymbol{\eta}), \quad \boldsymbol{\eta} \sim p(\boldsymbol{\eta}), \tag{6}$$

where $\boldsymbol{\theta}_j$ and $\boldsymbol{\eta}$ are called the local and global parameters, respectively.

Example: Population Dynamics

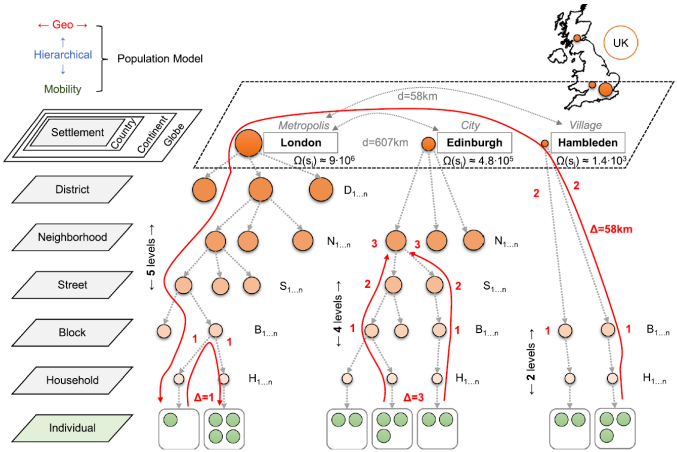
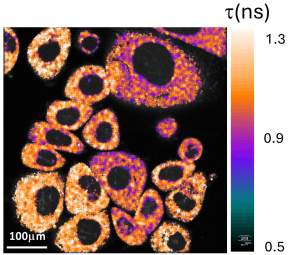
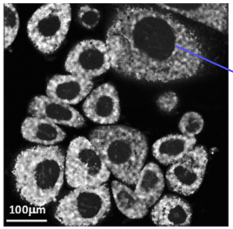


Figure from Topirceanu and Precup (2021).

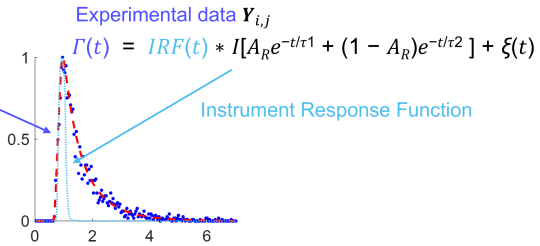
Example: Fluorescence Lifetime Imaging



Fluorescence lifetime



Spatially resolved fluorescence decays



Fluorescence decay at one pixel

Straightforward Amortization (I)

- The task of Bayesian estimation is to estimate the full joint posterior over local and global parameters:

$$p(\boldsymbol{\eta}, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_J \mid \mathbf{Y}_1, \ldots, \mathbf{Y}_J) \propto p(\boldsymbol{\eta}) \prod_{j=1}^{J} p(\mathbf{Y}_j \mid \boldsymbol{\theta}_j) p(\boldsymbol{\theta}_j \mid \boldsymbol{\eta}), \tag{7}$$

- A non-unique inverse factorization of the posterior is:

$$p(\boldsymbol{\eta}, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_J \mid \mathbf{Y}_1, \ldots, \mathbf{Y}_J) = p(\boldsymbol{\eta} \mid \{\mathbf{Y}_j\}) \prod_{j=1}^{J} p(\boldsymbol{\theta}_j \mid \mathbf{Y}_j, \boldsymbol{\eta}) \tag{8}$$

- Hence, a direct amortization objective is:

$$\min_{q_1, q_2} \mathbb{E}_{p(\boldsymbol{\eta}, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_J, \mathbf{Y}_1, \ldots, \mathbf{Y}_J)} \left[-\log q_1(\boldsymbol{\eta} \mid \{\mathbf{Y}_j\}) - \sum_{j=1}^{J} \log q_2(\boldsymbol{\theta}_j \mid \mathbf{Y}_j, \boldsymbol{\eta}) \right] \tag{9}$$

Straightforward Amortization (II)

- Problems: The number of groups J and number of observations per group N_j can vary, so q_1 and q_2 must see and process an arbitrary number of conditions.
- Hence, the actual amortization objective is

$$\min_{q_1,q_2,h_1,h_2} \mathbb{E}_{p(J)p(\boldsymbol{\eta},\boldsymbol{\theta}_{1:J},\mathbf{Y}_{1:J})} \left[-\log q_1(\boldsymbol{\eta} \mid h_1(\{h_2(\mathbf{Y}_j)\})) - \sum_{j=1}^{J} \log q_2(\boldsymbol{\theta}_j \mid h_2(\mathbf{Y}_j), \boldsymbol{\eta}) \right], \tag{10}$$

  where h_1 and h_2 are local (e.g., per-group) and global (e.g., per-experiment) embedding backbones, respectively.
- Already for $J = 1000$, generating a single training instance can take up to a few hours.

Example Two-Level Architecture

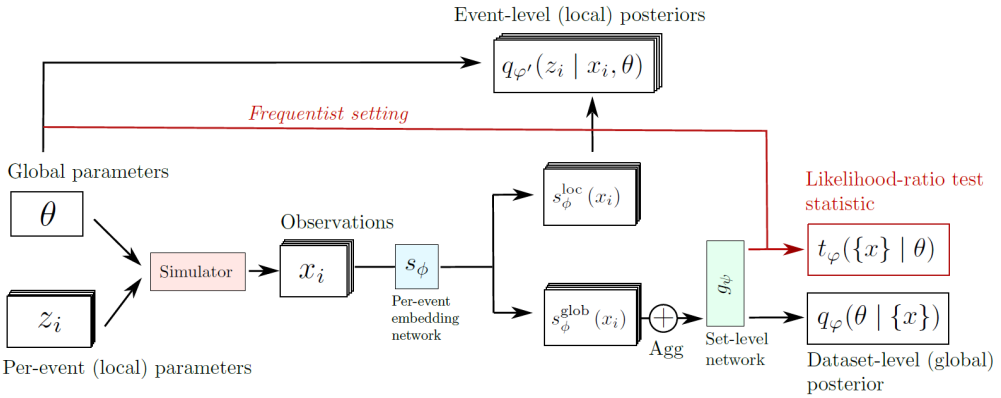
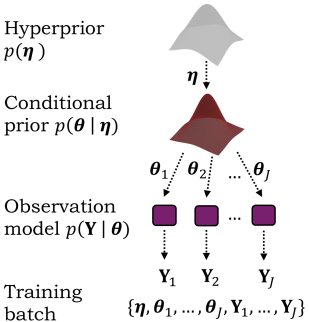


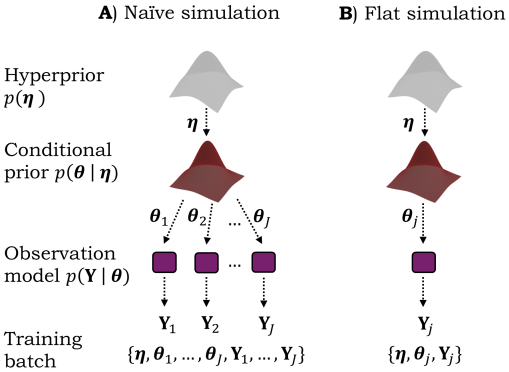
Figure from Heinrich et al. (2024).

A Divide-and-Conquer Approach

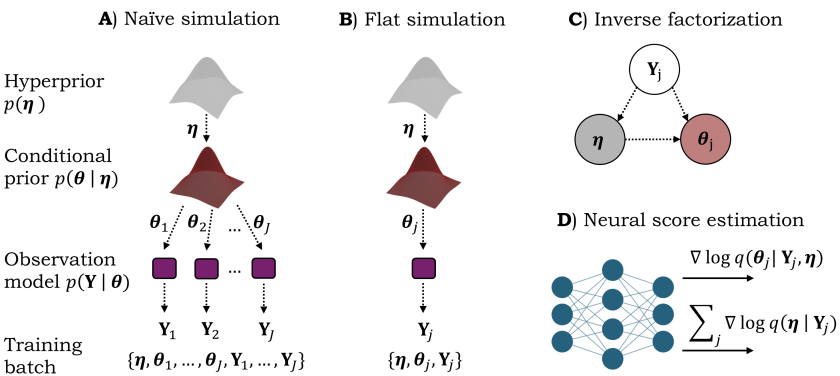
A) Naïve simulation



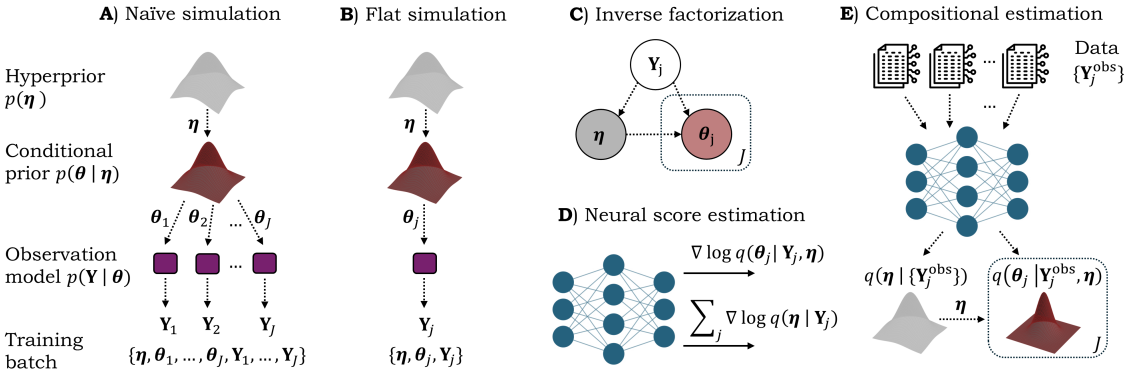
A Divide-and-Conquer Approach



A Divide-and-Conquer Approach



A Divide-and-Conquer Approach



Denoising Diffusion Models: Forward SDE (I)

- Diffusion models treat target generation as a denoising process (Song & Ermon, 2019; Ho et al., 2020):
- Starting with a target sample $\mathbf{z}_0 \equiv \boldsymbol{\theta}$, the "noising" process at time point t (with initial time $t = 0$ and final time $t = 1$) performs iterations of the form

$$\mathbf{z}_{t+\Delta t} = \mathbf{z}_t + f_t\,\mathbf{z}_t\,\Delta t + g_t \boldsymbol{\epsilon} \quad \text{with} \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{11}$$

- With $\Delta t \to 0$, we get a stochastic differential equation (SDE):

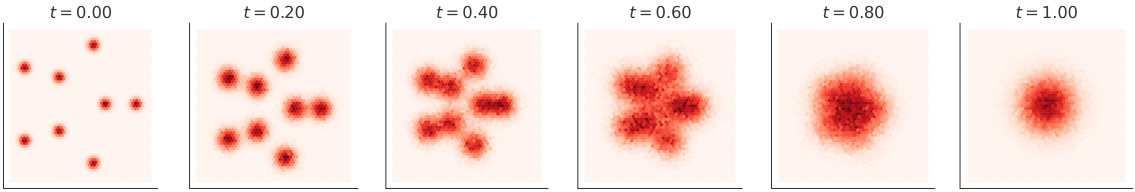
$$\mathrm{d}\mathbf{z}_t = f(t)\,\mathbf{z}_t\,\mathrm{d}t + g(t)\,\mathrm{d}\mathbf{W}. \tag{12}$$

- The conditional distribution of states when we evolve $\mathbf{z}_0$ to $\mathbf{z}_t$ is:

$$p(\mathbf{z}_t \mid \mathbf{z}_0) = \mathcal{N}(\alpha_t\,\mathbf{z}_0,\, \sigma_t^2\,\mathbf{I}) \quad \Longleftrightarrow \quad \mathbf{z}_t = \alpha_t\,\mathbf{z}_0 + \sigma_t\,\boldsymbol{\epsilon} \quad \text{with} \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{13}$$

Denoising Diffusion Models: Forward SDE (II)

Forward "noising" process destroying the structure of an 8-mode Gaussian Mixture Model (GMM):



Denoising Diffusion Models: Reverse SDE (I)

- To generate the target from noise, we run the SDE in reverse. The reverse SDE has the same form as the forward with negative time steps $\mathrm{d}\tilde{t} < 0$:

$$\mathrm{d}\mathbf{z}_t = \widetilde{f}(t, \mathbf{z}_t)\,\mathrm{d}\tilde{t} + g(t)\,\mathrm{d}\mathbf{W}. \tag{14}$$

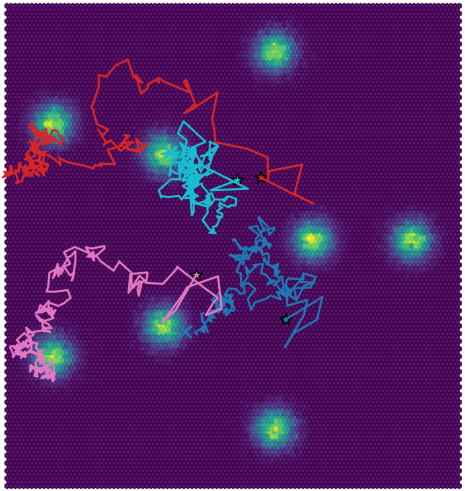
- The new drift terms can also be expressed analytically (Anderson, 1982):

$$\widetilde{f}(t, \mathbf{z}_t) = f(t)\,\mathbf{z}_t - g_t^2\ \nabla_{\mathbf{z}_t} \log p(\mathbf{z}_t \mid \mathbf{z}_0). \tag{15}$$

- The quantity $\nabla_{\mathbf{z}_t} \log p(\mathbf{z}_t \mid \mathbf{z}_0)$ is called the score of the conditional distribution. It ensures that the denoising SDE indeed returns to the starting point $\mathbf{z}_0$ of the corresponding "noising" SDE.

Denoising Diffusion Models: Reverse SDE (II)

- Memo: The score term $\nabla_{\mathbf{z}_t} \log p(\mathbf{z}_t \mid \mathbf{z}_0)$ steers the trajectories toward high-density regions of p^*, recovering structure from noise.
- The figure depicts four sample trajectories obtained with an "oracle" Euler-Maruyama solver (star = start, cross = target).



Denoising Score Matching

- Score matching learns a neural network $s(\mathbf{z}_t, t) \approx \nabla_{\mathbf{z}_t} \log p(\mathbf{z}_t \mid \mathbf{z}_0)$ that does not depend on the potentially unknown $\mathbf{z}_0$ for evaluation.
- The training objective for the network is a version of:

$$\min_s \mathbb{E}_{\mathbf{z}_0, \mathbf{z}_t, t \sim p(\mathbf{z}_0, \mathbf{z}_t, t)} \left[\omega_t \left\| s(\mathbf{z}_t, t) - \nabla_{\mathbf{z}_t} \log p(\mathbf{z}_t \mid \mathbf{z}_0) \right\|_2^2 \right] \tag{16}$$

- Memo: Since $p(\mathbf{z}_t \mid \mathbf{z}_0)$ is Gaussian and $\mathbf{z}_t = \alpha_t \mathbf{z}_0 + \sigma_t \boldsymbol{\epsilon}$, we have:

$$\min_s \mathbb{E}_{\mathbf{z}_0 \sim p^*(\mathbf{z}_0), t \sim \mathcal{U}(0,1), \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})} \left[\omega_t \left\| s(\alpha_t \, \mathbf{z}_0 + \sigma_t \, \boldsymbol{\epsilon}, t) + \boldsymbol{\epsilon} / \sigma_t \right\|_2^2 \right] \tag{17}$$

- Technical note: We parameterize s with the log-SNR, $\lambda_t = \log(\alpha_t / \sigma_t)$. To make the model conditional, simply insert $\mathbf{Y}$ along $\mathbf{z}_t$, hence the score has the form $s(\mathbf{z}_t, \mathbf{Y}, \lambda_t)$.

The Fun Starts Here: Compositional Score Matching

- For J exchangeable groups of data points, $\{\mathbf{Y}_j\}_{j=1}^J$, the compositional posterior for $\boldsymbol{\eta}$ can be written as

$$p(\boldsymbol{\eta} \mid \{\mathbf{Y}_j\}_{j=1}^J) \propto p(\boldsymbol{\eta})^{1-J} \prod_{j=1}^J p(\boldsymbol{\eta} \mid \mathbf{Y}_j), \tag{18}$$

- Define the time-varying bridging densities (Geffner et al., 2023):

$$p_t(\boldsymbol{\eta}_t \mid \{\mathbf{Y}_j\}_{j=1}^J) \propto p(\boldsymbol{\eta}_t)^{(1-J)(1-t)} \prod_{j=1}^J p_t(\boldsymbol{\eta}_t \mid \mathbf{Y}_j) \tag{19}$$

- We get a linear composition of the prior score and individual posterior scores:

$$\nabla_{\boldsymbol{\eta}_t} \log p_t(\boldsymbol{\eta}_t \mid \{\mathbf{Y}_j\}_{j=1}^J) = (1-J)(1-t)\nabla_{\boldsymbol{\eta}_t} \log p(\boldsymbol{\eta}_t) + \sum_{j=1}^J \nabla_{\boldsymbol{\eta}_t} \log p_t(\boldsymbol{\eta}_t \mid \mathbf{Y}_j). \tag{20}$$

Unfortunately...This Doesn't Work

- Trivial analytic non-hierarchical model for $\boldsymbol{\eta} \in \mathbb{R}^{10}$ and $\mathbf{Y} = \{\mathbf{y}_j\}$

$$\boldsymbol{\eta} \sim \mathcal{N}(\boldsymbol{\eta} \mid \mathbf{0}, \xi^2 \mathbf{I}) \tag{21}$$

$$\mathbf{y}_j \sim \mathcal{N}(\mathbf{y} \mid \boldsymbol{\eta}, \sigma^2 \mathbf{I}) \quad \text{for} \quad j = 1, \ldots, J \tag{22}$$

Convergence of sampling methods with a maximum budget of $10,000$ compositional sampling steps and maximal 30 minutes of runtime for a single $\mathbf{Y}$ (✓– converges, ✗– fails).

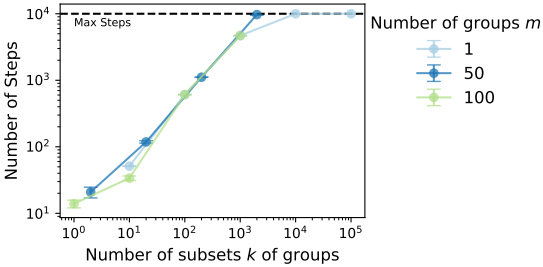
Method	J=10	J=100	J=10k	J=100k
Annealed Langevin sampler (Geffner et al., 2023)	✓	✗	✗	✗
`GAUSS` (Linhart et al., 2024)	✓	✓	✗	✗

What About Adaptive Solvers?

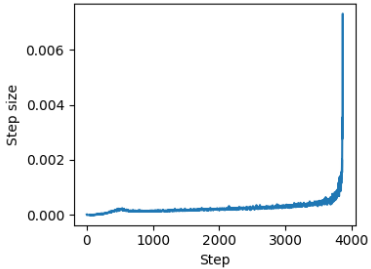
Convergence of sampling methods with a maximum budget of $10,000$ compositional sampling steps and maximal 30 minutes of runtime for a single $\mathbf{Y}$ (✓– converges, ✗– fails).

Method	J=10	J=100	J=10k	J=100k
Annealed Langevin sampler (Geffner et al., 2023)	✓	✗	✗	✗
`GAUSS` (Linhart et al., 2024)	✓	✓	✗	✗
Euler-Maruyama sampler	✓	✗	✗	✗
Probability ODE sampler	✓	✓	✗	✗
Adaptive second-order sampler	✓	✓	✗	✗

Adaptive Solver Behavior



(a) Number of sampling steps.



(b) Adaptive step size for 100 groups.

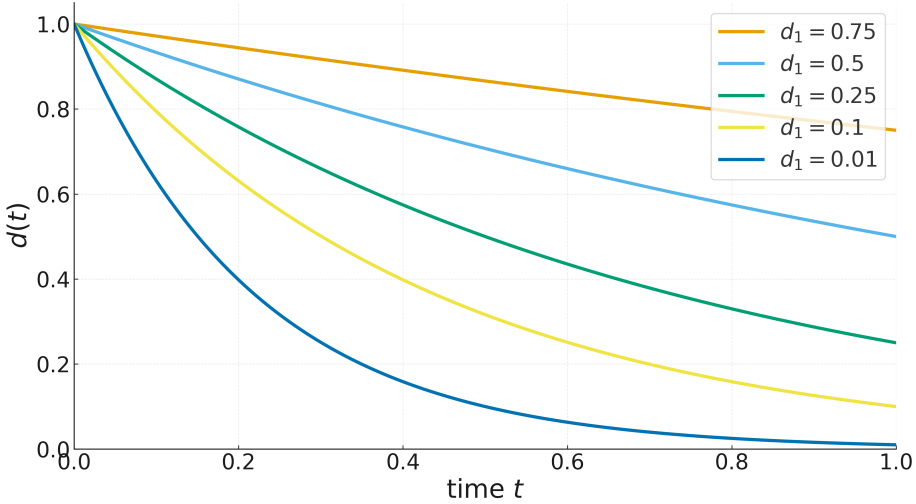
Stabilizing the Composition: Error-Damping

- Key idea: Define a monotonic function $d(t)$ that modulates the accumulation of score contributions throughout the diffusion trajectory during inference.
- This leads to a more flexible family of bridging densities:

$$p_t(\boldsymbol{\eta}_t \mid \{\mathbf{Y}_j\}_{j=1}^J) \propto \left(p(\boldsymbol{\eta}_t)^{1-J}\right)^{(1-t)d(t)} \prod_{j=1}^{J} p_t(\boldsymbol{\eta}_t \mid \mathbf{Y}_j)^{d(t)}, \tag{23}$$

  with $d(0) = d_0 = 1$ and $d(1) = d_1 \leq 1$.
- We use exponential decay $d(t) = d_0 \cdot \exp(-\log(d_0/d_1) \cdot t)$ with $d_0 = 1$ and a hyperparameter d_1 that can be tuned during inference.

Error-Damping Schedule



Scaling the Composition: Mini-Batch Estimation

- To address memory constraints in big-data settings, we introduce a mini-batch estimator for the compositional score:

$$\hat{s}(\boldsymbol{\eta}_t, \{\mathbf{Y}_j\}_{j=1}^{J}, \lambda_t) = (1-J)(1-t)\nabla_{\boldsymbol{\eta}_t} \log p(\boldsymbol{\eta}_t) + \frac{J}{M}\sum_{i=1}^{M} s(\boldsymbol{\eta}_t, \mathbf{Y}_{j_i}, \lambda_t), \tag{24}$$

  where $j_i \sim \mathcal{U}\{1, \ldots, J\}$ and M is the mini-batch size.
- The mini-batch estimator in Eq. 24 is an unbiased estimator of the compositional score (for a short proof, see Arruda et al., 2025).

Final Compositional Estimator

- Combining the mini-batch estimator with the damping function gives our final compositional estimator:

$$\hat{s}^d(\boldsymbol{\eta}_t, \{\mathbf{Y}_j\}_{j=1}^J, \lambda_t) = d(t) \cdot \left((1-J)(1-t)\nabla_{\boldsymbol{\eta}_t} \log p(\boldsymbol{\eta}_t) + \frac{J}{M} \sum_{i=1}^{M} s(\boldsymbol{\eta}_t, \mathbf{Y}_{j_i}, \lambda_t) \right), \quad (25)$$

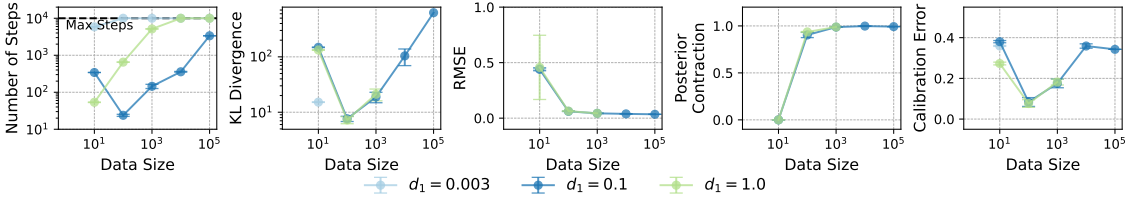
  where $j_i \sim \mathcal{U}\{1, \ldots, J\}$ and M is the mini-batch size.
- This estimator trades off some posterior error for extreme scalability.
- Bonus: Use cosine noise schedule $\lambda(t) = -2 \cdot \log(\tan(\pi t/2)) + 2s$ for larger step sizes during inference.

Gaussian Toy Example Summary (I)

Convergence of sampling methods with a maximum budget of $10,000$ compositional sampling steps and maximal 30 minutes of runtime for a single $\mathbf{Y}$ (✓– converges, ✗– fails).

Method	J=10	J=100	J=10k	J=100k
Annealed Langevin sampler (Geffner et al., 2023)	✓	✗	✗	✗
`GAUSS` (Linhart et al., 2024)	✓	✓	✗	✗
Euler-Maruyama sampler	✓	✗	✗	✗
Probability ODE sampler	✓	✓	✗	✗
Adaptive second-order sampler	✓	✓	✗	✗
Any sampler with damping (ours)	✓	✓	✓	✓
Any sampler with schedule adjustment (ours)	✓	✓	✓	✓

Gaussian Toy Example Summary (II)



- Key observation 1: A simple rule of thumb $d_1 \approx 1/\sqrt{J}$ successfully composes 10^5 scores, keeping RMSE negligible.
- Key observation 2: Mini-batching lowers per-step cost and, at $\approx 10\%$, achieves low calibration error at a slight RMSE cost (see paper for a gazillion plots).

Divide-and-Conquer Hierarchical Estimation

- Training: learn the following score models jointly:

$$s^{\text{global}}(\boldsymbol{\eta}_t, \mathbf{Y}_j, \lambda_t) \approx \nabla_{\boldsymbol{\eta}_t} \log p_t(\boldsymbol{\eta}_t \mid \mathbf{Y}_j) \tag{26}$$

$$s^{\text{local}}(\boldsymbol{\theta}_{t,j}, \mathbf{Y}_j, \boldsymbol{\eta}, \lambda_t) \approx \nabla_{\boldsymbol{\theta}_t} \log p_t(\boldsymbol{\theta}_{t,j} \mid \boldsymbol{\eta}, \mathbf{Y}_j) \tag{27}$$

- Sampling: use ancestral sampling:

$$\boldsymbol{\eta} \sim q^{\text{global}}(\boldsymbol{\eta} \mid \{\mathbf{Y}_j\}_{j=1}^{J}) \qquad \text{[compositional]} \tag{28}$$

$$\boldsymbol{\theta}_j \sim q^{\text{local}}(\boldsymbol{\theta} \mid \mathbf{Y}_j, \boldsymbol{\eta}) \qquad \text{[standard diffusion]} \tag{29}$$

with an optional embedding backbone for $\mathbf{Y}_j$, if needed.

Experiment: Hierarchical AR Model (1)

- Hyperpriors for global parameters:

$$\alpha \sim \mathcal{N}(0,1), \quad \beta \sim \mathcal{N}(0,1), \quad \log \sigma \sim \mathcal{N}(0,1). \tag{30}$$

- The local parameters are different for each grid point:

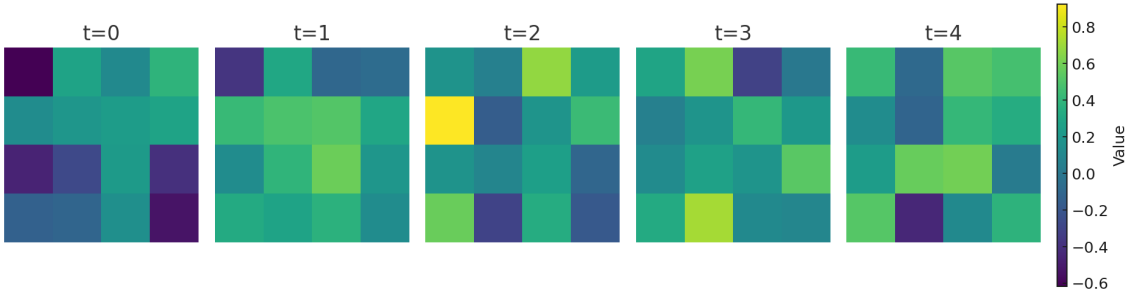
$$\tilde{\theta}_j \sim \mathcal{N}(0, \sigma^2), \qquad \theta_j = 2\,\mathrm{sigmoid}(\beta + \tilde{\theta}_j) - 1. \tag{31}$$

- For each grid point j, we have a time series of $T = 5$ observations,

$$\begin{aligned} y_{j,0} &\sim \mathcal{N}(0, \sigma = 0.1) \\ y_{j,t} &\sim \mathcal{N}(\alpha + \theta_j y_{j,t-1}, \sigma = 0.1), \quad t = 1, \dots T-1. \end{aligned}$$

Experiment: Hierarchical AR Model (2)

- Example realization for $J = 16$:



Experiment: Hierarchical AR Model (3)

- Excerpt of benchmarking results against NUTS (breakeven point at 32 x 32):

Method	RMSE ($\boldsymbol{\eta}$)	Contraction ($\boldsymbol{\eta}$)	RMSE (θ)	Contraction (θ)
NUTS (4x4)	0.08 (0.05)	0.95 (0.04)	0.1 (0.01)	0.98 (0.00)
Ours (4x4)	0.09 (0.05)	0.97 (0.0)	0.14 (0.01)	0.95 (0.01)
NUTS (128x128)	0.01 (0.01)	1.0 (0.00)	0.09 (0.03)	0.99 (0.00)
Ours (128x128)	0.09 (0.05)	1.0 (0.01)	0.13 (0.01)	0.97 (0.01)

- Total simulation budget amounts to 610 grids of size 128×128!

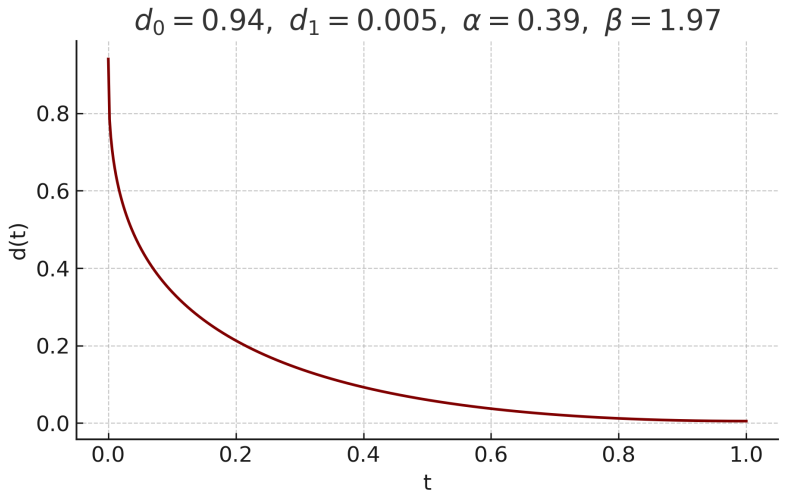
Bonus: Inference-Time Hyperparameter Optimization

- Introduce a more flexible damping function:

$$d(t) = d_0 + (d_1 - d_0) \cdot \left(1 - (1 - t^{\alpha})^{\beta}\right) \tag{32}$$

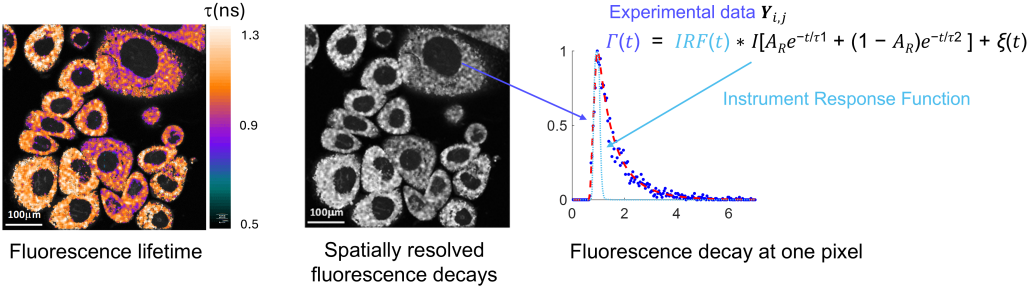
- The hyperparameters α and β enable smooth interpolation between linear, exponential-like, and cosine-like behaviors.
- Perform Bayesian optimization (BO) (e.g., using `optuna`) over the hypercube $\mathcal{P} = [\alpha_{\min}, \alpha_{\max}] \times [\beta_{\min}, \beta_{\max}] \times [d_{1,\min}, d_{1,\max}] \times [d_{0,\min}, d_{0,\max}]$.
- Use a posterior metric as an objective (e.g., a weighted combination of RMSE and Calibration Error).

Bonus: Inference-Time Hyperparameter Optimization



Experiment: Fluorescence Lifetime Imaging (1)

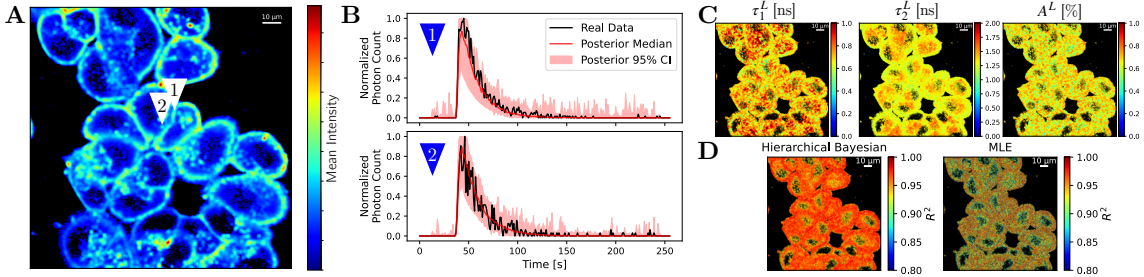
- Goal: Fluorescence Lifetime Imaging (FLI) under photon-limited noise in:



- Model: Each pixel in 512×512 images is modeled with a bi-exponential decay, with local parameters $(\tau_1^L, \tau_2^L, A^L)$ and global priors over their means and variances.
- Training: Training requires only single-pixel simulations (the equivalent of only 350 full images), rather than full-image simulations.

Experiment: Fluorescence Lifetime Imaging (2)

- Fully Bayesian hierarchical modeling on $J = 262,144$ (!) decay time-series for a total of $750,000$ parameters:



A Mean pixel intensity over time; **B** Pixel time series with posterior median fit; **C** Spatial map of posterior medians; **D** Spatial R^2 map vs. RMLE baseline (*ours*: 0.961, *baseline*: 0.871).

Concluding Remarks

- Our method plugs into modern diffusion models and enables simulation-efficient hierarchical Bayesian inference by composing individual scores during inference.
- Error-damping bridging densities + mini-batch score aggregation prevent divergence and control error accumulation in the reverse SDE.
- Damping schedules and noise-schedule shifts can be optimized post-training (e.g., via Bayesian Optimization) to balance contraction and calibration (or any set of metrics).
- The method works for hundreds of thousands of groups.

The End

Agenda

Discussion

- Get a grip on calibration: How to achieve good calibration under extreme contraction (e.g., importance sampling)?
- Generalize to non-exchangeable hierarchical models using ideas from Gloeckler et al. (2024)?
- Explore non-linear score aggregation strategies (e.g., non-linear links)?
- Refine mini-batch selection using informativeness criteria (e.g., typicality sampling, Peng et al., 2019)?
- Combine with MCMC whenever (differentiable) likelihoods are available?

References I

Anderson, B. D. (1982). Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, *12*(3), 313–326.

Arruda, J., Pandey, V., Sherry, C., Barroso, M., Intes, X., Hasenauer, J., & Radev, S. T. (2025). Compositional amortized inference for large-scale hierarchical bayesian models. *arXiv preprint arXiv:2505.14429*.

Elsemüller, L., Olischläger, H., Schmitt, M., Bürkner, P.-C., Koethe, U., & Radev, S. T. (2024). Sensitivity-aware amortized bayesian inference. *Transactions on Machine Learning Research*.

Geffner, T., Papamakarios, G., & Mnih, A. (2023). Compositional score modeling for simulation-based inference. In *International conference on machine learning* (pp. 11098–11116).

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., & Rubin, D. B. (2013). *Bayesian data analysis (3rd edition)*. Chapman and Hall/CRC.

Gloeckler, M., Toyota, S., Fukumizu, K., & Macke, J. H. (2024). Compositional simulation-based inference for time series. *arXiv preprint arXiv:2411.02728*.

References II

Heinrich, L., Mishra-Sharma, S., Pollard, C., & Windischhofer, P. (2024). Hierarchical neural simulation-based inference over event ensembles. *arXiv preprint arXiv:2306.12584*.

Ho, J., Jain, A., & Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in neural information processing systems*, *33*, 6840–6851.

Linhart, J., Cardoso, G. V., Gramfort, A., Corff, S. L., & Rodrigues, P. L. (2024). Diffusion posterior sampling for simulation-based inference in tall data settings. *arXiv preprint arXiv:2404.07593*.

McElreath, R. (2018). *Statistical rethinking: A Bayesian course with examples in r and stan*. Chapman and Hall/CRC.

Peng, X., Li, L., & Wang, F.-Y. (2019). Accelerating minibatch stochastic gradient descent using typicality sampling. *IEEE transactions on neural networks and learning systems*, *31*(11), 4649–4659.

Schmitt, M., Li, C., Vehtari, A., Acerbi, L., Bürkner, P.-C., & Radev, S. T. (2024). Amortized bayesian workflow. *arXiv preprint arXiv:2409.04332*.

References III

Song, Y., & Ermon, S. (2019). Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, *32*.

Topirceanu, A., & Precup, R.-E. (2021). A novel geo-hierarchical population mobility model for spatial spreading of resurgent epidemics. *Scientific Reports*, *11*(1), 14341.