Bayesian Learning via Neural Schrödinger-Föllmer Flows

Francisco Vargas[1*], Andrius Ovsianas[2], David Fernandes[4], Mark Girolami[2], Neil D. Lawrence[1] and Nikolas Nüsken[4]

[1*]Department of Computer Science, Cambridge University, Cambridge, CB3 0FD, UK.
[2]Department of Engineering, Cambridge University, Cambridge, CB2 1PZ, UK.
[3]Department of Computer Science, University Of Bath, Bath, Bath BA2 7PB, UK.
[4]Institute of Mathematics, University of Potsdam, Potsdam, 14476, Germany.

*Corresponding author(s). E-mail(s): fav25@cam.ac.uk;
Contributing authors: ao464@cam.ac.uk; dlf28@bath.ac.uk; mag92@cam.ac.uk; ndl21@cam.ac.uk; nuesken@uni-potsdam.de;

Abstract

In this work we explore a new framework for approximate Bayesian inference in large datasets based on stochastic control. We advocate stochastic control as a finite time and low variance alternative to popular steady-state methods such as stochastic gradient Langevin dynamics (SGLD). Furthermore, we discuss and adapt the existing theoretical guarantees of this framework and establish connections to already existing VI routines in SDE-based models.

Keywords: Schrödinger Bridge Problem, Föllmer Drift, Stochastic Control, Bayesian Inference, Bayesian Deep Learning.

1 Introduction

Steering a stochastic flow from one distribution to another across the space of probability measures is a well-studied problem initially proposed in Schrödinger [65]. There has been recent interest in the machine learning community in these methods for generative modelling, sampling, dataset imputation and optimal transport [4, 10, 12, 15, 39, 52, 61, 70, 72].

We consider a particular instance of the Schrödinger bridge problem (SBP), known as the Schrödinger-Föllmer process (SFP). In machine learning, this process has been proposed for sampling and generative modelling [39, 69] and in molecular dynamics for rare event simulation and importance sampling [31, 32]; here we apply it to Bayesian inference. We show that a control-based formulation of the SFP has deep-rooted connections to variational inference and is particularly well suited to Bayesian inference in high dimensions. This capability arises from the SFP's characterisation as an optimisation problem and its parametrisation through neural networks [69]. Finally, due to the variational characterisation that these methods possess, many low-variance estimators [54, 62, 64, 74] are applicable to the SFP formulation we consider.

We reformulate the Bayesian inference problem by constructing a stochastic process $\boldsymbol{\Theta}_t$ which at a fixed time $t = 1$ will generate samples from a pre-specified posterior $p(\boldsymbol{\theta}|\boldsymbol{X})$, i.e. Law$\boldsymbol{\Theta}_1 = p(\boldsymbol{\theta}|\boldsymbol{X})$, with dataset $\boldsymbol{X} = \{\mathbf{x}_i\}_{i=1}^N$, and where the

Published at Statistics and Computing, 2022.

model is given by:

$$\begin{aligned} \boldsymbol{\theta} &\sim p(\boldsymbol{\theta}), \\ \mathbf{x}_i|\boldsymbol{\theta} &\sim p(\boldsymbol{x}_i|\boldsymbol{\theta}). \quad \text{iid} \end{aligned} \tag{1}$$

Here the prior $p(\boldsymbol{\theta})$ and the likelihood $p(\mathbf{x}_i|\boldsymbol{\theta})$ are user-specified. Our target is $\pi_1(\boldsymbol{\theta}) = \frac{p(\boldsymbol{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\mathcal{Z}}$, where $\mathcal{Z} = \int \prod_i p(\boldsymbol{x}_i|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}$. This formulation is reminiscent of the setup proposed in the previous works [23, 28, 63, 73] and covers many Bayesian machine-learning models, but our formulation has an important difference. SGLD relies on a diffusion that reaches the posterior as its equilibrium state when time approaches infinity. In contrast, our dynamics are *controlled* and the posterior is reached in finite time (bounded time). The benefit of this property is elegantly illustrated in Section 3.2 of [39] where they rigorously demonstrate that even under an Euler approximation the proposed approach reaches a Gaussian target at time $t = 1$ whilst SGLD does not.

Contributions: The main contributions of this work can be detailed as follows:

- In this work we scale and apply the theoretical framework proposed in [13, 68] to sample from posteriors in large scale Bayesian machine learning tasks such as Bayesian Deep learning. We study the robustness of the predictions under this framework as well as evaluate their uncertainty quantification.
- More precisely we propose an amortised parametrisation that allows scaling models with local and global variables to large datasets.
- We explore and provide further theoretical backing (Section 2.2) to the "sticking the landing" estimator provided by [74].
- Overall we empirically demonstrate that the stochastic control framework offers a promising direction in Bayesian machine learning, striking the balance between theoretical/asymptotic guarantees found in MCMC methods [9, 18, 33, 53] and more practical approaches such as variational inference [7].

1.1 Notation

Throughout the paper we consider path measures (denoted as $\mathbb{Q}$ or $\mathbb{S}$) on the space of continuous functions $\Omega = C([0,1], \mathbb{R}^d)$. Random processes associated with such path measures $\mathbb{Q}$ are denoted as $\boldsymbol{\Theta}$ and their time-marginal distributions as $\mathbb{Q}_t = (\boldsymbol{\Theta}_t)_{\#}\mathbb{Q}$ (which are just pushforward measures). Given two marginal distributions π_0 and π_1 we write $\mathcal{D}(\pi_0, \pi_1) = \{\mathbb{Q} : \mathbb{Q}_0 = \pi_0, \mathbb{Q}_1 = \pi_1\}$ for the set of all path measures with given marginal distributions at the initial and final times. We denote by $\mathbb{Q}^{\mathbf{u},\pi}$ the path measure of the following Stochastic Differential Equation (SDE):

$$d\boldsymbol{\Theta}_t = \mathbf{u}(t, \boldsymbol{\Theta}_t)dt + \sqrt{\gamma}d\mathbf{B}_t, \quad \boldsymbol{\Theta}_0 \sim \pi \tag{2}$$

(we drop the dependence on γ since it is fixed) and we write $\mathbb{W}^\gamma = \mathbb{Q}^{0,\delta_0}$ for the Wiener measure. We will write $\frac{d\mathbb{Q}}{d\mathbb{S}}$ for the Radon-Nikodym derivative (RND) of $\mathbb{Q}$ w.r.t. $\mathbb{S}$.

1.2 Schrödinger-Föllmer Processes

Definition 1 *(Schrödinger-Bridge Process) Given a reference process $\mathbb{S}$ and two measures π_0 and π_1 the Schrödinger bridge distribution is given by*

$$\mathbb{Q}^* = \underset{\mathbb{Q}\in\mathcal{D}(\pi_0,\pi_1)}{\arg\inf} D_{\mathrm{KL}}\left(\mathbb{Q}\middle|\middle|\mathbb{S}\right), \tag{3}$$

where $\mathbb{S}$ acts as a "prior".

It is known [50] that if $\mathbb{S} = \mathbb{Q}^{\mathbf{u},\pi}$, $\mathbb{Q}^*$ is induced by an SDE with a modified drift:

$$\mathrm{d}\boldsymbol{\Theta}_t = \mathbf{u}^*(t, \boldsymbol{\Theta}_t)\mathrm{d}t + \sqrt{\gamma}\mathrm{d}\mathbf{B}_t, \quad \boldsymbol{\Theta}_0 \sim \pi_0, \tag{4}$$

i.e. $\mathbb{Q}^* = \mathbb{Q}^{\mathbf{u}^*,\pi_0}$. Solution of this SDE is called the Schrödinger-Bridge Process (SBP).

Definition 2 *(Schrödinger-Föllmer Process) The SFP is an SBP where $\pi_0 = \delta_0$ and the reference process $\mathbb{S} = \mathbb{W}^\gamma$ is the Wiener measure.*

The SFP differs from the general SBP in that, rather than constraining the initial distribution to δ_0, the SBP considers *any* initial distribution π_0. The SBP also involves general Itô SDEs associated with $\mathbb{Q}^{\boldsymbol{u},\pi}$ as the dynamical prior, compared to the SFP which restricts attention to Wiener processes as priors.

The advantage of considering this more limited version of the SBP is that it admits a closed-form characterisation of the solution to the Schrödinger system [50, 58, 72] which allows for an unconstrained formulation of the problem. For accessible introductions to the SBP we suggest [58, 70]. Now

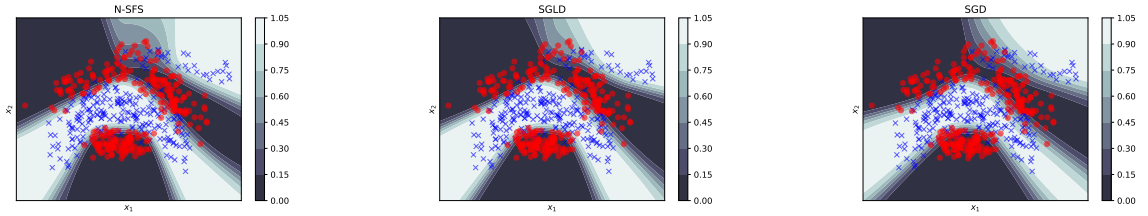

Fig. 1 Predictive posterior contour plots on the banana dataset [16]. Test accuracies: $0.8928 \pm 0.0056, 0.8913 \pm 0.0105, 0.8800 \pm 0.0063$ and test ECEs: $0.0229 \pm 0.0062, 0.0253 \pm 0.0042, 0.0267 \pm 0.0083$ for N-SFS, SGLD, and SGD respectively. We observe that N-SFS obtains the highest test accuracy whilst preserving the lowest ECE.

we will consider instances of the SBP and the SFP where $\pi_1 = p(\boldsymbol{\theta}|\boldsymbol{X})$.

1.2.1 Analytic Solutions and the Heat Semigroup

Prior work [13, 39, 57, 69] has explored the properties of SFPs via a closed form formulation of the Föllmer drift expressed in terms of expectations over Gaussian random variables known as the heat semigroup. The seminal works [13, 57, 69] highlight how this formulation of the Föllmer drift characterises an exact sampling scheme for a target distribution and how it could potentially be used in practice. The recent work by [39] builds on [69] and explores estimating the optimal drift in practice via the heat semigroup formulation using a Monte Carlo approximation. Our work aims to take the next step and scale the estimation of the Föllmer drift to high dimensional cases [27, 37]. In order to do this we must move away from the heat semigroup and instead consider the dual formulation of the Föllmer drift in terms of a stochastic control problem [69].

In the setting when $\pi_0 = \delta_0$ we can express the optimal SBP drift as follows

$$\mathbf{u}^*(t, \boldsymbol{x}) = \nabla_{\boldsymbol{x}} \ln \mathbb{E}_{\boldsymbol{\Theta} \sim \mathbb{S}} \left[\frac{d\pi_1}{d\mathbb{S}_1}(\boldsymbol{\Theta}_1) \middle| \boldsymbol{\Theta}_t = \boldsymbol{x} \right] \quad (5)$$

Definition 3 *The Euclidean heat semigroup* Q_t^γ, $t \geq 0$, *acts on bounded measurable functions* $f : \mathbb{R}^d \to \mathbb{R}$ *as* $Q_t^\gamma f(\boldsymbol{x}) = \int_{\mathbb{R}^d} f\left(\boldsymbol{x} + \sqrt{t}\boldsymbol{z}\right) \mathcal{N}(\boldsymbol{z}|\mathbf{0}, \gamma\mathbb{I}) d\boldsymbol{z} = \mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \gamma\mathbb{I})} \left[f\left(\boldsymbol{x} + \sqrt{t}\boldsymbol{z}\right) \right]$.

In the SFP case where $\mathbb{S} = \mathbb{W}^\gamma$, the optimal drift from Equation 5 can be written in terms of the heat semigroup, $\mathbf{u}^*(t, \boldsymbol{x}) = \nabla_{\boldsymbol{x}} \ln Q_{1-t}^\gamma \left[\frac{d\pi_1}{d\mathcal{N}(\mathbf{0}, \gamma\mathbb{I})}(\boldsymbol{x}) \right]$. Note that an SDE with the heat semigroup induced drift

$$d\boldsymbol{\Theta}_t = \nabla_{\boldsymbol{\Theta}_t} \ln Q_{1-t}^\gamma \left[\frac{d\pi_1}{d\mathcal{N}(\mathbf{0}, \gamma\mathbb{I})}(\boldsymbol{\Theta}_t) \right] dt + \sqrt{\gamma} d\mathbf{B}_t \quad (6)$$

satisfies $\text{Law}\boldsymbol{\Theta}_1 = \pi_1$, that is, at $t = 1$ these processes are distributed according to our target distribution of interest π_1.

1.2.2 Schrödinger-Föllmer Samplers

[39] carried out preliminary work on empirically exploring the success of using the heat semigroup formulation of SFPs in combination with the Euler-Mayurama (EM) discretisation to sample from target distributions in a method they call Schrödinger-Föllmer samplers (SFS). More precisely the SFS approach proposes estimating the Föllmer drift via:

$$\hat{\mathbf{u}}^*(t, \boldsymbol{x}) = \frac{\frac{1}{S} \sum_{s=1}^S \boldsymbol{z}_s f(\boldsymbol{x} + \sqrt{1-t}\boldsymbol{z}_s)}{\frac{\sqrt{1-t}}{S} \sum_{s=1}^S f(\boldsymbol{x} + \sqrt{1-t}\boldsymbol{z}_s)}, \quad (7)$$

where $\boldsymbol{z}_s \sim \mathcal{N}(\mathbf{0}, \gamma\mathbb{I})$ and $f = \frac{d\pi_1}{d\mathcal{N}(\mathbf{0}, \gamma\mathbb{I})}$. Whilst this estimator enjoys sound theoretical properties [39] it falls short in practice for the following reasons:

- The term f involves the product of PDFs evaluated at samples rather than a log product and is thus often very unstable numerically. In Appendix F we provide a more stable implementation of Equation 7 exploiting the logsumexp trick and properties of the Lebesgue integral.
- In it's current form the estimator does not admit low variance estimators (e.g. Variatonal Inference), being a Monte Carlo estimator it is prone to high variance.

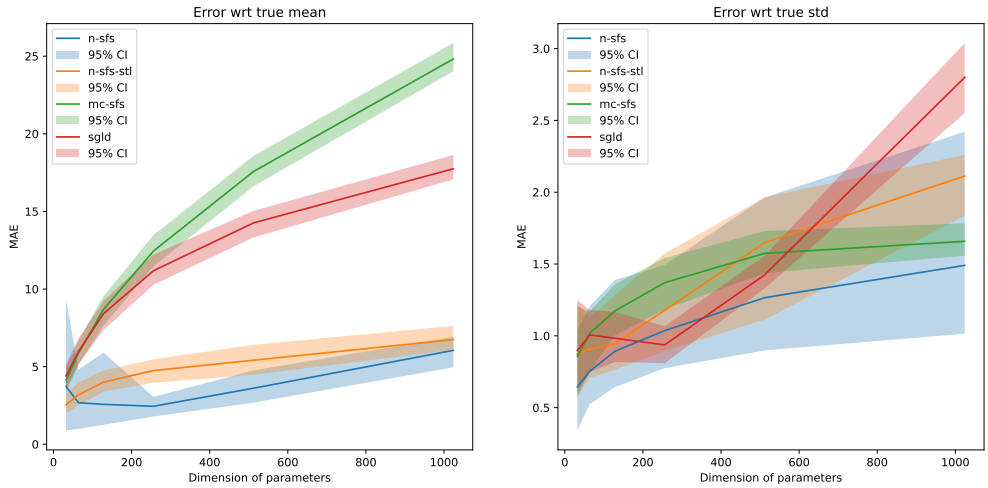

Fig. 2 Comparison between MC-SFS and N-SFS under similar computational constraints. Target distribution is the Gaussian posterior induced by a Bayesian linear regression model, we plot the error of the first and second posterior predictive moments between the true posterior predictive and the listed approximations. We found increasing the number of steps in SGLD drove the errors closer to 0 however when increasing the dimensions this threshold also increased notably. This illustrates the advantages of having a target at a finite time rather than at equilibrium.

- Both empirically and theoretically we found the computational running time of the above approach to be considerably slower than the other methods we compare to. At test time SFS has a computational complexity of $\mathcal{O}(TS\#_f(d))$ where $T = \Delta t^{-1}$, S is the number of Monte Carlo samples and $\#_f(d)$ is the cost of evaluating the RND f which at best is linear in d. Meanwhile our proposed approach enjoys a cost of $\mathcal{O}(T\#_{\boldsymbol{u}_\phi}(d))$ where $\#_{\boldsymbol{u}_\phi}(d)$ is the forward pass through a neural network approximating the Föllmer drift.

In practice we found this implementation to be too numerically unstable and unable to produce reasonable results even in low dimensional examples in order to carry out a fair comparison we reformulated Equation 7 stably, the stable formulation and its derivation can be found in Appendix F.

In this work build on [39] by considering a formulation of the Schrödinger-Föllmer process that is suitable for the high dimensional settings arising in Bayesian ML. Our work will focus on a dual formulation of the optimal drift that is closer to variational inference and thus admits the scalable and flexible parametrisations used in ML.

2 Stochastic Control Formulation

In this section, we introduce a particular formulation of the Schrödinger-Föllmer process in the context of the Bayesian inference problem in Equation 1. In its most general setting of sampling from a target distribution, this formulation was known to [13]. [69] study the theoretical properties of this approach in the context of generative models [24, 43], finally [55] applies this formulation to time series modelling. In contrast our focus is on the estimation of a Bayesian posterior for a broader class of models than Tzen and Raginsky explore.

Corollary 1 *Define*

$$\mathcal{F}_{\mathrm{DET}}(\mathbf{u}, \boldsymbol{\theta}) = \frac{1}{2\gamma} \int_0^1 \|\mathbf{u}(t, \boldsymbol{\theta}_t)\|^2 dt - \ln \frac{p(\boldsymbol{X}|\boldsymbol{\theta}_1)p(\boldsymbol{\theta}_1)}{\mathcal{N}(\boldsymbol{\theta}_1|\mathbf{0}, \gamma \mathbb{I}_d)}$$

$$J(\mathbf{u}) = \mathbb{E}_{\boldsymbol{\Theta} \sim \mathbb{Q}^{\mathbf{u}, \delta_0}} \left[\mathcal{F}_{\mathrm{DET}}(\mathbf{u}, \boldsymbol{\Theta})\right]$$

Algorithm 1 Optimization of N-SFS with Stochastic Mini-batches.

Input: data set $\boldsymbol{X} = \{\mathbf{x}_i\}_{i=1}^N$, initialized drift NN $\boldsymbol{u}_\phi$, parameter dimension d, # of iterations M, batch size B, # of EM discretization steps k, # of MC samples S , diffusion coefficient γ.
Initialise: $\Delta t \leftarrow \frac{1}{k}$, $t_j \leftarrow j\Delta t$ for all $j = 0, \dots, k$
for $i = 1, \dots, M$ **do**
  Initialize $\boldsymbol{\Theta}_0^s \leftarrow 0 \in \mathbb{R}^d$ for all $s = 1, \dots, S$
  $\{\Theta_j^{s\phi}\}_{j=1}^k \leftarrow$ Euler-Maruyama$(\boldsymbol{u}_\phi, \boldsymbol{\Theta}_0^s, \Delta t)$ for all $s = 1, \dots, S$
  Sample $\mathbf{x}_{r_1}, \dots, \mathbf{x}_{r_B} \sim \boldsymbol{X}$
  $g \leftarrow \nabla_\phi \left(\frac{1}{S} \sum_{s=1}^{S} \sum_{j=0}^{k} \left(||\boldsymbol{u}_\phi(\boldsymbol{\Theta}_j^{s\phi}, t_j)||^2 \Delta t - \ln\left(\frac{p(\boldsymbol{\Theta}_k^{s\phi})}{\mathcal{N}(\boldsymbol{\Theta}_k^{s\phi} | \mathbf{0}, \gamma \mathbb{I}_d)} \right) + \frac{N}{B} \sum_{j=1}^{B} \ln p(\mathbf{x}_{r_j} | \boldsymbol{\Theta}_k^{s\phi}) \right) \right)$
  $\phi \leftarrow$ Gradient Step(ϕ, g)
end for
Return: $\boldsymbol{u}_\phi$

Then the minimiser (with $\mathcal{U}$ being the set of admissible controls[1])

$$\mathbf{u}^* = \arg\min_{\mathbf{u} \in \mathcal{U}} J(\mathbf{u}) \tag{8}$$

satisfies $\mathbb{Q}_1^{\gamma, \mathbf{u}^*, \delta_0} = \frac{p(\boldsymbol{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\mathcal{Z}} d\boldsymbol{\theta}$.
Moreover, $\mathbf{u}^*$ solves the SFP with $\pi_1 = p(\boldsymbol{\theta}|\boldsymbol{X})$.

The objective in Equation 8 can be estimated using an SDE discretisation, such as the EM method. Since the drift $\mathbf{u}^*$ is Markov, it can be parametrised by a flexible function estimator such as a neural network, as in [69]. In addition, unbiased estimators for the gradient of objective in equation 8 can be formed by subsampling the data. In this work we will refer to the above formulation of the SFP as the Neural Schrödinger-Föllmer sampler (N-SFS) when we parametrise the drift with a neural network and implement unbiased mini-batched estimators for this objective (Appendix C). This formulation of SFPs has been previously studied in the context of generative modelling / marginal likelihood estimation [69], while we focus on Bayesian inference.

We note that recent concurrent work [78] [2] proposes an algorithm akin to ours based on [13, 69], however their focus is on estimating the normalising constant of unnormalised densities, while ours is on Bayesian ML tasks such as Bayesian regression, classification and LVMs, thus our work leads to different insights and algorithmic motivations.

[1] Under appropriate conditions on the model in Equation 1, $\mathcal{U}$ can be taken to be the set of C^1-vector fields with linear growth in space, see [54].

[2] This work was made public on arxiv within a month of our arxiv pre-print release.

2.1 Theoretical Guarantees for Neural SFS

While the focus in [69] is in providing guarantees for generative models of the form $\boldsymbol{x} \sim q_\phi(\boldsymbol{x}|\boldsymbol{Z}_1)$, $d\boldsymbol{Z}_t = \boldsymbol{u}_\phi(\boldsymbol{Z}_t, t)dt + \sqrt{\gamma} d\mathbf{B}_t$, $\boldsymbol{Z}_0 = 0$, their results extend to our setting as they explore approximating the Föllmer drift for a generic target π_1.

Theorem 4 in Tzen and Raginsky (restated as Theorem 2 in Appendix A.2) motivates using neural networks to parametrise the drift in Equation 8 as it provides a guarantee regarding the expressivity of a network parametrised drift via an upper bound on the target distribution error in terms of the size of the network.

We will now proceed to highlight how this error is affected by the EM discretisation:

Corollary 2 *Given the network $\boldsymbol{v}$ from Theorem 2 it follows that the Euler-Maruyama discretisation of equation 2 with $\mathbf{u} = \boldsymbol{v}$ induces an approximate target $\hat{\pi}_1^{\boldsymbol{v}}$ that satisfies*

$$D_{\mathrm{KL}}(\pi_1 || \hat{\pi}_1^{\boldsymbol{v}}) \leq \left(\epsilon^{1/2} + \mathcal{O}(\sqrt{\Delta t}) \right)^2 . \tag{9}$$

This result provides a bound of the error in terms of the depth Δt^{-1} of the stochastic flow [10, 77] and the size of the network that we parametrise the drift with. Under the view that NN parametrised SDEs can be interpreted as ResNets [51] we find that this result illustrates that increasing the ResNets' depth will lead to more accurate results.

2.2 Sticking the Landing and Low Variance Estimators

As with VI [62, 64], the gradient of the objective in this study admits several low variance estimators [54, 74]. In this section we formally recap what it means for an estimator to "stick the landing" and we prove that the estimator proposed in Xu et al. satisfies said property.

The full objective being minimised in our approach is (where expectations are taken over $\boldsymbol{\Theta} \sim \mathbb{Q}^{\mathbf{u},\delta_0}$):

$$\begin{aligned} J(\mathbf{u}) &= \mathbb{E}[\mathcal{F}_{\mathrm{DET}}(\mathbf{u},\boldsymbol{\Theta})] \\ &= \mathbb{E}[\mathcal{F}(\mathbf{u},\boldsymbol{\Theta})] \\ &= \mathbb{E}\left[\frac{1}{2\gamma}\int_0^1 ||\mathbf{u}_t(\boldsymbol{\Theta}_t)||^2 dt + \frac{1}{\sqrt{\gamma}}\int_0^1 \mathbf{u}_t(\boldsymbol{\Theta}_t)^\top d\mathbf{B}_t \right. \\ &\qquad \left. -\ln\left(\frac{p(\boldsymbol{X}|\boldsymbol{\Theta}_1)p(\boldsymbol{\Theta}_1)}{\mathcal{N}(\boldsymbol{\Theta}_1|\mathbf{0},\gamma\mathbb{I}_d)}\right)\right], \end{aligned} \tag{10}$$

noticing that in previous formulations we have omitted the Itô integral as it has zero expectation (but the integral appears naturally through Girsanov's theorem). We call the estimator calculated by taking gradients of the above objective the relative-entropy estimator. The estimator proposed in [74] (Sticking the landing estimator) is given by:

$$\begin{aligned} J_{\mathrm{STL}}(\mathbf{u}) &= \mathbb{E}[\mathcal{F}_{\mathrm{STL}}(\mathbf{u},\boldsymbol{\Theta})] \\ &= \mathbb{E}\left[\frac{1}{2\gamma}\int_0^1 ||\mathbf{u}_t(\boldsymbol{\Theta}_t)||^2 dt + \frac{1}{\sqrt{\gamma}}\int_0^1 \mathbf{u}_t^{\perp}(\boldsymbol{\Theta}_t)^\top d\mathbf{B}_t \right. \\ &\qquad \left. -\ln\left(\frac{p(\boldsymbol{X}|\boldsymbol{\Theta}_1)p(\boldsymbol{\Theta}_1)}{\mathcal{N}(\boldsymbol{\Theta}_1|\mathbf{0},\gamma\mathbb{I}_d)}\right)\right], \end{aligned} \tag{11}$$

where $\perp$ means that the gradient is stopped/detached as in [64, 74].

We study perturbations of $\mathcal{F}$ around $\mathbf{u}^*$ by considering $\mathbf{u}^* + \varepsilon\boldsymbol{\phi}$, with $\boldsymbol{\phi}$ arbitrary, and ε small. More precisely, we set out to compute (where dependence on $\boldsymbol{\theta}$ is dropped):

$$\frac{\mathrm{d}}{\mathrm{d}\varepsilon}\mathcal{F}(\mathbf{u}^* + \varepsilon\boldsymbol{\phi})\Big|_{\varepsilon=0}, \tag{12}$$

through which we define the definition of "sticking the landing":

Definition 4 *We say that an estimator "sticks the landing" when*

$$\frac{\mathrm{d}}{\mathrm{d}\varepsilon}\mathcal{F}(\mathbf{u}^* + \varepsilon\boldsymbol{\phi})\Big|_{\varepsilon=0} = 0, \tag{13}$$

almost surely, for all smooth and bounded perturbations $\boldsymbol{\phi}$.

Notice that by construction, $\boldsymbol{u}^*$ is a global minimiser of J, and hence all directional derivatives vanish,

$$\frac{\mathrm{d}}{\mathrm{d}\varepsilon}J(\mathbf{u}^* + \varepsilon\boldsymbol{\phi})\Big|_{\varepsilon=0} = \frac{\mathrm{d}}{\mathrm{d}\varepsilon}\mathbb{E}[\mathcal{F}(\mathbf{u}^* + \varepsilon\boldsymbol{\phi},\boldsymbol{\Theta})]\Big|_{\varepsilon=0} = 0. \tag{14}$$

Definition 4 additionally demands that this quantity is zero almost surely, and not just on average. Consequently, "sticking the landing"-estimators will have zero-variance at $\mathbf{u}^*$.

Remark 1 *The relative-entropy stochastic control estimator does not stick the landing.*

Proof See [54], Theorem 5.3.1, clause 3, Equation 133 clearly indicates $\frac{\mathrm{d}}{\mathrm{d}\varepsilon}\mathcal{F}(\mathbf{u}^* + \varepsilon\boldsymbol{\phi})\Big|_{\varepsilon=0} \neq 0$. □

We can now go ahead and prove that the estimator proposed by [74] does indeed stick the landing.

Theorem 1 *The STL estimator proposed in [74] satisfies*

$$\frac{\mathrm{d}}{\mathrm{d}\varepsilon}\mathcal{F}(\mathbf{u}^* + \varepsilon\boldsymbol{\phi})\Big|_{\varepsilon=0} = 0, \tag{15}$$

almost surely, for all smooth and bounded perturbations $\boldsymbol{\phi}$.

The proof for the above result can be found in Appendix E, and combines results from [54].

2.3 Structured SVI in Models with Local and Global Variables

Algorithm 1 produces unbiased estimates of the gradient[3] as demonstrated in Appendix C only under the assumption that the parameters are

[3] Gradients are computed automatically via reverse mode differentiation [2, 22] using the pytorch library [56].

global, that is when there is not a local parameter for each data point. In the setting where we have local and global variables we can no longer do mini-batch updates as in Algorithm 1 since the energy term in the objective does not decouple as a sum over the datapoints [36, 37]. In this section we discuss said limitation and propose a reasonable heuristic to overcome it.

We consider the general setting where our model has global and local variables $\boldsymbol{\Phi}, \{\boldsymbol{\theta}_i\}$ satisfying $\boldsymbol{\theta}_i \perp\!\!\!\perp \boldsymbol{\theta}_j | \boldsymbol{\Phi}$ [37]. This case is particularly challenging as the local variables scale with the size of the dataset and so will the state space. This is a fundamental setting as many hierachical latent variable models in machine learning admit such dependancy structure, such as Topic models [7, 60]; Bayesian factor analysis [1, 6, 14, 45]; Variational GP Regression [35]; and others.

Remark 2 *The heat semigroup does not preserve conditional independence structure in the drift, i.e. the optimal drift does not decouple and thus depends on the full state-space (Appendix D).*

Remark 2 tells us that the drift is not structured in a way that admits scalable sampling approaches such as stochastic variational inference (SVI) [37]. Additionally this also highlights that the method by [39] does not scale to models like this as the dimension of the state space will be linear in the size of the dataset.

In a similar fashion to Hoffman and Blei [36], who focussed on structured SVI, we suggest parametrising the drift via $[\mathbf{u}_t]_{\boldsymbol{\theta}_i} = u^{\boldsymbol{\theta}_i}(t, \boldsymbol{\theta}_i, \Phi, \boldsymbol{x}_i)$; this way the dimension of the drift depends only on the respective local variables and the global variable Φ. While the Föllmer drift does not admit this particular decoupling we can show that this drift is flexible enough to represent fairly general distributions, thus it is expected to have the capacity to reach the target distribution. Via this parametrisation we can sample in the same fashion as SVI and maintain unbiased gradient estimates.

Remark 3 *An SDE parametrised with a decoupled drift* $[\mathbf{u}_t]_{\boldsymbol{\theta}_i} = u^{\boldsymbol{\theta}_i}(t, \boldsymbol{\theta}_i, \Phi, \boldsymbol{x}_i)$ *can reach transition densities which do not factor (See Appendix D for proof).*

It is important to highlight that whilst the parametrisation in Remark 3 may be flexible, it may not satisfy the previous theory developed for the Föllmer drift and SBPs, thus an interesting direction would be in recasting the SBP such that the optimal drift is decoupled. However, we found in practice that the decoupled and amortised drift worked very well, outperforming SGLD and the non-decoupled N-SFS.

3 Connections Between SBPs and Variational Inference in Latent Diffusion Models

In this section, we highlight the connection between the objective in Equation 8 to variational inference in models with an SDE as the latent object, as studied in [68]. We first start by making the connection in a simpler case – when the prior of our Bayesian model is given by a Gaussian distribution with variance γ, that is $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\mathbf{0}, \gamma \mathbb{I}_d)$.

Observation 1 *When* $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\mathbf{0}, \gamma\mathbb{I}_d)$*, it follows that the N-SFP objective in Equation 8 corresponds to the negative ELBO of the model:*

$$\begin{aligned} d\boldsymbol{\Theta}_t = \sqrt{\gamma} d\mathbf{B}_t, \quad \boldsymbol{\Theta}_0 \sim \delta_0, \\ \mathbf{x}_i \sim p(\mathbf{x}_i|\boldsymbol{\Theta}_1). \end{aligned} \tag{16}$$

While the above observation highlights a specific connection between N-SFP and traditional VBI (Variational Bayesian Inference), it is limited to Bayesian models that are specified with Gaussian priors. In Lemma 1 of Appendix B we extend this result to more general priors and reference process via exploiting the general recursive nature of Bayesian updates [42]. In short, we can view the objective in Equation 8 as an instance of variational Bayesian inference with an SDE prior. Note that this provides a succinct connection between variational inference and maximum entropy in path space [49]. In more detail, this observation establishes an explicit connection between the ELBO of an SDE-based generative model where the SDE is latent and the SBP/stochastic-control objectives we explore in this work.

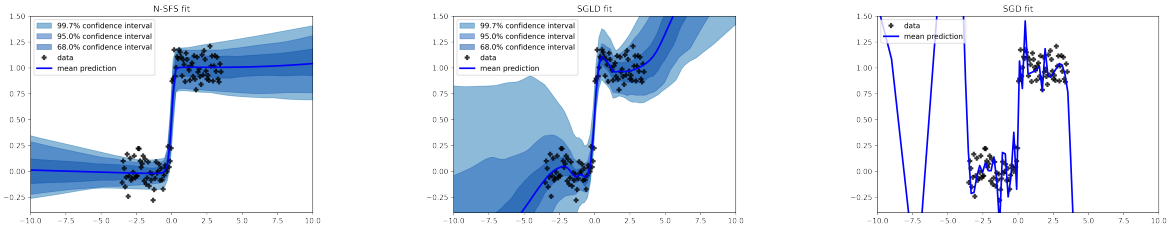

Fig. 3 Visual comparison on step function data. We can see how the N-SFS based fits have the best generalisation while SGD and SGLD interpolate the noise.

Table 1 a9a dataset.

Method	Accuracy	ECE	Log Likelihood
N-SFS	0.8498 ± 0.0002	0.0099 ± 0.0010	-0.3407 ± 0.0004
SGLD	0.8515 ± 0.0010	0.0010 ± 0.0020	-0.3247 ± 0.0002

Table 2 Step function dataset.

Method	MSE	Log Likelihood
N-SFS	0.0028 ± 0.0010	-63.048 ± 8.2760
SGLD	0.1774 ± 0.1280	-1389.581 ± 834.9680

Table 3 MEG dataset.

Method	Log Likelihood
N-SFS	-5.1110 ± 0.1288
SGLD	-4.9360 ± 0.0423

Note that Lemma 1 induces a new two stage algorithm in which we first estimate a prior reference process as in Equation B10 and then we optimise the ELBO for the model in Equation B11. This raises the question as to what effect the dynamical prior can have within SBP-based frameworks. In practice we do not explore this formulation as the Föllmer drift of the prior may not be available in closed form and thus may require resorting to additional approximations.

4 Experimental Results

We ran experiments on Bayesian NN regression, classification, logistic regression and ICA [1], reporting accuracies, log joints [40, 73] and expected calibration error (ECE) [29]. For details on exact experimental setups please see Appendix H. Across experiments we compare to SGLD as it has been shown to be a competitive baseline in Bayesian deep learning [40]. Notice that we do not compare to more standard MCMC methodologies [17, 18, 53] as they do not scale well to very high dimensional tasks such as Bayesian DL [40] which are central to our experiments. However, [39] contrasts the performance of the heat semigroup SFS sampler with more traditional MCMC samplers in 2D toy examples, finding SFS to be competitive [4].

[4]Supporting code at https://github.com/franciscovargas/ControlledFollmerDrift.

4.1 Bayesian Linear Regression and Comparison with MC-SFS

In this section we explore a bayesian linear regression model with a prior on the regression weights. As this model has a Gaussian closed form for the posterior predictive distribution we report the error of the MC-SFS and N-SFS posterior predictive mean and variance with respect to the true posterior predictive moments as is seen in Figure 2. The datasets where generated by sampling the inputs randomly from a spherical Gaussian distribution and transforming them via:

$$y_i = \mathbf{1}^\top \boldsymbol{x}_i + 1$$

we then estimated the posterior of the model:

$$\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, \sigma_\theta^2 \mathbb{I}),$$
$$y_i | \boldsymbol{x}_i, \boldsymbol{\theta} \sim \mathcal{N}(y_i | \boldsymbol{\theta}^\top (\boldsymbol{x}_i \oplus 1), \sigma_y^2 \mathbb{I}),$$

Where we use $\boldsymbol{x} \oplus 1$ to denote adding an extra dimension with a 1 to the vector $\boldsymbol{x}$. We carried out this experiment increasing the dimension of $\boldsymbol{x}$ from 2^5 to 2^{11}. We observe that the N-SFS based approaches have overall a notably smaller posterior predictive error to the MC-SFS approach. Finally we note the STL method is more concentrated in its predictions than the naive N-SFS approach, whilst having similar errors.

4.2 Bayesian Logistic Regression / Independent Component Analysis - a9a / MEG Datasets

Following [73] we explore a logistic regression model on the a9a dataset. Results can be found in Table 1 which show that N-SFS achieves a test accuracy, ECE and log likelihood comparable to SGLD. We then explore the performance of our approach on the Bayesian variant of ICA studied in [73] on the MEG-Dataset [71]. We can observe (Table 3) that here N-SFS also achieves results comparable to SGLD.

4.3 Bayesian Deep Learning

In these tasks we use models of the form

$$\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, \sigma_\theta^2 \mathbb{I}),$$
$$\boldsymbol{y}_i | \boldsymbol{x}_i, \boldsymbol{\theta} \sim p(\boldsymbol{y}_i | f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)),$$

where $f_{\boldsymbol{\theta}}$ is a neural network. In these settings we are interested in using the posterior predictive distribution $p(\boldsymbol{y}^*|\boldsymbol{x}^*, \boldsymbol{X}) = \int p(\boldsymbol{y}^*|f_{\boldsymbol{\theta}}(\boldsymbol{x}^*))dP(\boldsymbol{\theta}|\boldsymbol{X})$ to make robust predictions. Across the image experiments we use the LeNet5 [47] architecture. Future works should explore recent architectures for images such as VGG-16 [66] and ResNet32 [34].

Non-linear Regression - Step Function: We fit a 2-hidden-layer neural network with a total of 14876 parameters on a toy step function dataset. We can see in Figure 3 how both the SGD and SGLD fits interpolate the noise, whilst N-SFS has straight lines, thus both achieving a better test error and having well-calibrated error bars. We believe it is a great milestone to see how an over-parameterised neural network is able to achieve such well calibrated predictions.

Digits Classification - LeNet5: We train the standard LeNet5 [47] architecture (with 44426 parameters) on the MNIST dataset [48]. At test time we evaluate the methods on the MNIST test set augmented by random rotations of up to 30°[21]. Table 4 shows how N-SFS has the highest accuracy whilst obtaining the lowest calibration error among the considered methods, highlighting that our approach has the most well-calibrated and accurate predictions when considering a slightly perturbed test set. We highlight that LeNet5 falls into an interesting regime as the number of parameters is considerably less than the size of the training set, and thus we can argue it is not in the overparameterised regime. This regime [3] has been shown to be challenging in achieving good generalisation errors, thus we believe the predictive and calibrated accuracy achieved by N-SFS is a strong milestone.

Additionally we provide results on the regular MNIST test set. We can observe that N-SFS maintains a high test accuracy and at the same time preserves a low ECE score. We believe the reason SGD and SGLD obtain slightly better ECE performances is that the MNIST test set has very little variation to the MNIST training set, and thus all results seem well calibrated. We can see this observation confirmed by how the distribution of ECE scores changes dramatically on the Rotated MNIST set, a similar argument to that developed in [21]. We note that across both experiments SGLD achieves a slightly better log likelihood which comes at the cost of lower predictive performance and less calibrated predictions.

Image Classification - CIFAR10: We fit a variation of the LeNet5 (Appendix H.4) architecture with 62006 parameters on the CIFAR10 dataset [46]. We note that the predictive test accuracies and log-likelihoods of N-SFS$_{\text{stl}}$, SGLD and SGD are comparable. However, we can see that N-SFS$_{\text{stl}}$ has an ECE an order of magnitude smaller. We notice that the STL estimator made a significant difference on CIFAR10, making the training faster and more stable.

4.4 Hyperspectral Image Unmixing

To assess our method's performance visually, we use it to sample from Hyperspectral Unmixing Models [5]. Hyperspectral images are high spectral resolution but low spatial resolution images typically taken of vast areas via satellites. High spectral resolution provides much more information about the materials present in each pixel. However, due to the low spatial resolution, each pixel of an image can correspond to a $50m^2$ area, containing several materials. Such pixels will therefore have mixed and uninformative spectra. The task of Hyperspectral Unmixing is to determine the presence of given materials in each pixel.

Table 4 Test set results on MNIST, Rotated MNIST and CIFAR10. The Log-likelihood column is the mean posterior predictive and is thus not estimated for SGD.

Dataset	Method	Accuracy	ECE	Log Likelihood
MNIST	N-SFS	0.9889 ± 0.0013	0.0080 ± 0.0013	-0.0883 ± 0.0076
	$\text{N-SFS}_{\text{stl}}$	0.9885 ± 0.0014	0.0092 ± 0.0017	-0.0629 ± 0.0057
	SGLD	0.9837 ± 0.0007	0.0061 ± 0.0012	-0.0516 ± 0.0026
	SGD	0.9884 ± 0.0007	0.0034 ± 0.0009	-
Rotated-MNIST	N-SFS	0.9479 ± 0.0043	0.0077 ± 0.0012	-0.3890 ± 0.0374
	$\text{N-SFS}_{\text{stl}}$	0.9461 ± 0.0039	0.0057 ± 0.0012	-0.2960 ± 0.0336
	SGLD	0.9247 ± 0.0035	0.0141 ± 0.0018	-0.2439 ± 0.0118
	SGD	0.9404 ± 0.0031	0.0284 ± 0.0021	-
CIFAR10	N-SFS	0.6156 ± 0.0021	0.0520 ± 0.0110	-1.3628 ± 0.0262
	$\text{N-SFS}_{\text{stl}}$	0.6264 ± 0.0286	0.0568 ± 0.0069	-1.2305 ± 0.0710
	SGLD	0.6232 ± 0.0186	0.1493 ± 0.0170	-1.2740 ± 0.0854
	SGD	0.6229 ± 0.0124	0.0626 ± 0.0163	-

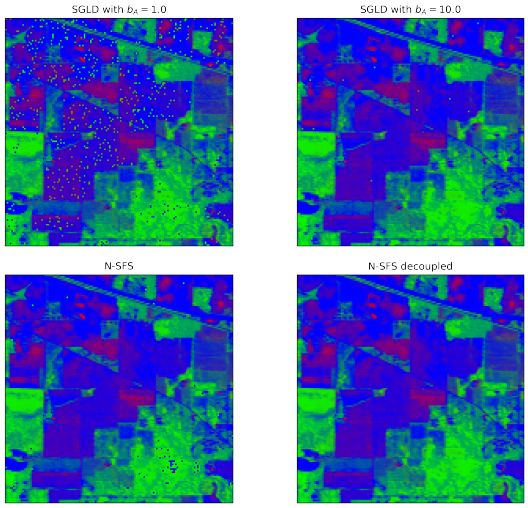

Fig. 4 False-color composites with channels given by the unmixed matrices $\boldsymbol{A}$ obtained via SGLD, N-SFS and N-SFS with a decoupled drift. Speckles illustrate mode collapse.

We use the Indian Pines image[5], denoted as $\boldsymbol{Y}$, which has a spatial resolution of $P = 145 \times 145 = 21025$ pixels and a spectral resolution of $B = 200$ bands, i.e. $\boldsymbol{Y} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_P] \in [0,1]^{B \times P}$. $R = 3$ materials have been chosen automatically using the Pixel Purity Index and the collection of their spectra will be denoted as $\boldsymbol{M} = [\boldsymbol{m}_1, \boldsymbol{m}_2, \boldsymbol{m}_3] \in [0,1]^{B \times 3}$. The task of Hyperspectral Unmixing is to determine for each pixel p a vector $\boldsymbol{a}_p \in \Delta_R$ in the probability simplex, where $[\boldsymbol{A}]_{p,i} = a_{p,i}$ represents the fraction of the i-th material in pixel p. To determine the presence of each material, we use the Normal Compositional Model [19] as it is a challenging model to sample from. Specifically, it has parameters $(\boldsymbol{\Phi}, \boldsymbol{\Theta}) = (\sigma^2, \boldsymbol{A})$ and is defined by:

$$p\left(\sigma^2\right) = \mathbf{1}_{[0,1]}\left(\sigma^2\right), \quad p\left(\boldsymbol{A}\right) = \prod_{p=1}^{P} \mathbf{1}_{\Delta_R}\left(\boldsymbol{a}_p\right),$$

$$p\left(\boldsymbol{Y}|\boldsymbol{A}, \sigma^2\right) = \prod_{p=1}^{P} \mathcal{N}\left(\boldsymbol{y}_p;\ \boldsymbol{M}\boldsymbol{a}_p;\ ||\boldsymbol{a}_p||^2 \sigma^2 I\right),$$

First note that this model follows the structured model setting discussed in Section 2.2 — it has one global parameter σ^2 and a local parameter $\boldsymbol{a}_p$ for each pixel. Finally, while all the parameters are constrained to lie on the probability simplices, this sampling problem can be cast into an unconstrained sampling problem via Lagrange transformations as in [38]. The Normal Compositional Model [19] is primarily of interest to us because the unusual noise scaling in the likelihood can produce several modes in each pixel, making it especially easy for sampling algorithms to get stuck in modes.

We compared three approaches for this problem: 1) SGLD 2) N-SFS 3) N-SFS with decoupled

[5] taken from http://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes

drift, where the decoupled drift is defined as:

$$\boldsymbol{u}_t(\sigma^2, \boldsymbol{A}) = [u_0(t, \sigma^2), u_1(t, \sigma^2, \boldsymbol{a}_1), \ldots, u_P(t, \sigma^2, \boldsymbol{a}_P)].$$

Unmixing results are shown in Figure 4. We stress that to run SGLD successfully we had to tune the approach heavily — we used separate step sizes (which acts as a preconditioning) and step size schedules for parameters σ^2 and $\boldsymbol{A}$, only with one combination of which we managed to get decent unmixing results. Without the amortised drift, N-SFS struggled with multiple modes in certain patches of the image, however, decoupling the drift resulted in almost perfect unmixing. With a slight deviation from the optimal step size schedule, SGLD fails to explore modes and produces speckly images. In contrast, the only tunable parameter for N-SFS was γ, which was giving similar results for all tried values. Further sensitivity results for SGLD/N-SFS are provided in Appendix G.

4.5 Analysis of N-SFS training dynamics

In addition to the experiments above, we investigate our method's performance in a synthetic multi-modal scenario. Here, N-SFS is used to fit a Gaussian Mixture posterior distribution that has modes aligned on the x-axis, as shown in figure 5. In one case, there are 4 modes – 2 inner modes (those closer to 0) and 2 outer modes (those further away from 0). We notice that in the presence of the 2 inner modes N-SFS is unable to discover the outer modes. In contrast, when considering a posterior with only the 2 outer modes, the distribution is fit correctly. This phenomenon could be explained by previously indicated connections between stochastic control and agent-based learning via the Hamilton-Jacobi-Bellman equation [59] and the exploration-exploitation tradeoff. More concretely, the optimisation objective equation 8 implies the following training dynamics – random samples are generated from a diffusion (a Brownian motion to begin with) which is then refined to produce more samples in areas where previous samples had high posterior density. This implies that after some modes are discovered, the diffusion will be adjusted to fit them, i.e. the algorithm immediately starts exploiting the detected modes. Other modes will only be discovered if some random sample accidentally hits them, which is very unlikely if the modes are far away. This indicates that the algorithm could be improved by incorporating exploration techniques found in agent-based learning literature.

Given the behaviour of N-SFS on this multi-modal example, it is then natural to ask if it happens in Bayesian Deep Learning applications. To examine this, we look at the marginal distributions of a pair of weights of a Bayesian Neural Network for MNIST classification given by the samples of N-SFS and SGLD given in Figure 6. Note that compared to SGLD, N-SFS samples from a dramatically wider distribution, while maintaining a comparable predictive log likelihood score, and therefore does not suffer from the lack of exploration.

5 Discussion and Future Directions

Overall we achieve predictive performance competitive to SGLD across a variety of tasks whilst obtaining better calibrated predictions as measured by the ECE metric. We hypothesise that the gain in performance is due to the flexible and low variance VI parametrisation of the proposed approach. We would like to highlight that these results were achieved with minimal tuning and simple NN architectures. We find that the decoupled and amortised drift we propose achieves very strong results making our approach tractable to Bayesian models with local and global structure. Additionally we notice that the architecture used in the drift network can influence results, thus future work in this area should develop the drift architectures further.

A key advantage of our approach is that at training time the objective effectively minimises an ELBO styled objective parameterised via a ResNet. This allows us to monitor training using the traditional techniques from deep learning, without the challenges arising from mixing times and correlation of samples found in traditional MCMC methods; once N-SFS is trained, generating samples at test time is a fast forward pass through a ResNet that does not require re-training. Finally, as we demonstrated, our approach allows the learned sampler to be amortised [76] which not only allows the drift

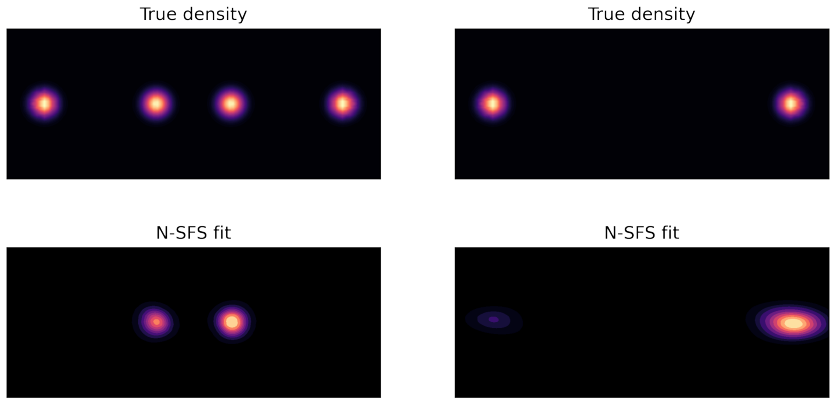

Fig. 5 N-SFS performance on a gaussian mixture posterior distribution with several modes. Outer modes are only detected when the posterior does not contain the interior modes indicating exploration failure of N-SFS.

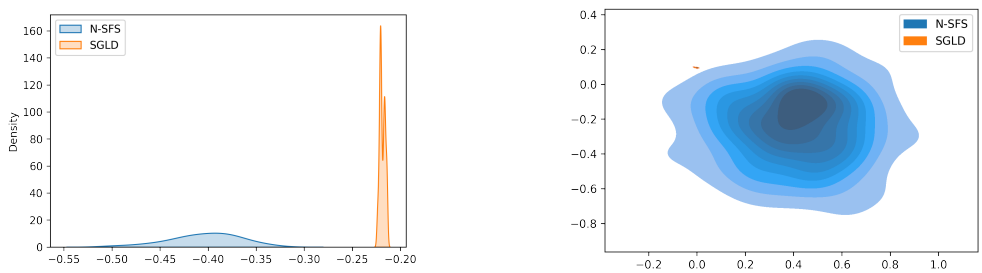

Fig. 6 Distribution of log posterior values of samples from N-SFS and SGLD **(left)** and marginal distribution of a pair of weights in a neural network obtained from samples of N-SFS and SGLD **(right)**

to be more tractably parameterised but also creates the prospects of meta learning the posterior [20, 25, 26, 75]. We believe that this work motivates how stochastic control paves a new exciting and promising direction in Bayesian ML/DL.

Acknowledgements. Francisco Vargas is Funded by Huawei Technologies Co. This research has been partially funded by Deutsche Forschungsgemeinschaft (DFG) through the grant CRC 1114 'Scaling Cascades in Complex Systems' (project A02, project number 235221301). Andrius Ovsianas is funded by EPSRC iCASE Award EP/T517677/1. Mark Girolami is supported by a Royal Academy of Engineering Research Chair, and EPSRC grants EP/T000414/1, EP/R018413/2, EP/P020720/2, EP/R034710/1, EP/R004889/1.

Appendix A Main Results

A.1 Posterior Drift

Corollary 1 *The minimiser*

$$\mathbf{u}^* = \underset{\mathbf{u}\in\mathcal{U}}{\arg\min}\, \mathbb{E}_{\boldsymbol{\Theta}\sim\mathbb{Q}^{\mathbf{u},\delta_0}} \left[\frac{1}{2\gamma}\int_0^1 \|\mathbf{u}(t,\boldsymbol{\Theta}_t)\|^2 dt - \ln\left(\frac{p(\boldsymbol{X}|\boldsymbol{\Theta}_1)p(\boldsymbol{\Theta}_1)}{\mathcal{N}(\boldsymbol{\Theta}_1|\mathbf{0},\gamma\mathbb{I}_d)}\right)\right] \tag{A1}$$

satisfies $\text{Law}\boldsymbol{\Theta}_1^{\mathbf{u}^*} = \frac{p(\boldsymbol{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\mathcal{Z}}$.

Proof This follows directly after substituting the Radon-Nikodym derivative between the Gaussian distribution and the posterior into Theorem 1 in [69] or Theorem 3.1 in [13]. □

A.2 EM-Discretisation Result

First we would like to introduce the following auxiliary theorem from [69]:

Theorem 2 *[69] Given the standard regularity assumptions presented for* $f = \frac{d\pi_1}{d\mathcal{N}(\mathbf{0},\gamma\mathbb{I})}$ *in [69], let* $L = \max\{\text{Lip}(f), \text{Lip}(\nabla f)\}$ *and assume that there exists a constant* $c \in (0,1]$ *such that* $f \geq c$. *Then for any* $\epsilon \in \left(0, 16\frac{L^2}{c^2}\right)$ *there exists a neural net* $\boldsymbol{v} : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d$ *with size polynomial in* $1/\epsilon, d, L, c, 1/c, \gamma$, *such that the activation function of each neuron follows the regularity assumptions in [69] (e.g.* ReLU, Sigmoid, Softplus*) and*

$$D_{\text{KL}}(\pi_1||\pi_1^{\boldsymbol{v}}) \leq \epsilon, \tag{A2}$$

where $\pi_1^{\boldsymbol{v}} = \text{Law}(\boldsymbol{\Theta}_1^{\boldsymbol{v}})$ *is the terminal distribution of the diffusion process*

$$d\boldsymbol{\Theta}_t^{\boldsymbol{v}} = \boldsymbol{v}(\boldsymbol{\Theta}_t^{\boldsymbol{v}}, \sqrt{1-t})dt + \sqrt{\gamma}d\mathbf{B}_t, \qquad t \in [0,1]. \tag{A3}$$

We can now proceed to prove the direct corollary of the above theorem when using the EM scheme for simulation.

Corollary 2 *Given the network* $\boldsymbol{v}$ *from Theorem 2 it follows that the Euler-Mayurama discretisation* $\hat{X}_t^{\boldsymbol{v}}$ *of* $X_t^{\boldsymbol{v}}$ *has a KL-divergence to the target distribution* π_1 *of:*

$$D_{\text{KL}}(\pi_1||\hat{\pi}_1^{\boldsymbol{v}}) \leq \left(\epsilon^{1/2} + \mathcal{O}(\sqrt{\Delta t})\right)^2 \tag{A4}$$

Proof Consider the path-wise KL-divergence between the exact Schrödinger-Föllmer process and its EM-discretised neural approximation:

$$D_{\text{KL}}(\mathbb{P}^{\mathbf{u}^*}||\mathbb{P}^{\hat{\boldsymbol{v}}}) = \frac{1}{2\gamma}\int_0^1 \mathbb{E}_{\boldsymbol{\Theta}\sim\mathbb{Q}^{\mathbf{u}*,\delta_0}} \left\|\mathbf{u}^*(\boldsymbol{\Theta}_t,t) - \hat{\boldsymbol{v}}(\boldsymbol{\Theta}_t,\sqrt{1-t})\right\|^2 dt. \tag{A5}$$

Defining $d(\boldsymbol{x},\boldsymbol{y}) := \sqrt{\frac{1}{2\gamma}\int_0^1 \mathbb{E}_{\boldsymbol{\Theta}\sim\mathbb{Q}^{\mathbf{u}*,\delta_0}} \|\boldsymbol{x}(\boldsymbol{\Theta}_t,t) - \hat{\boldsymbol{y}}(\boldsymbol{\Theta}_t,t)\|^2 dt}$, it is clear that $d(\boldsymbol{x},\boldsymbol{y})$ satisfies the triangle inequality as it is the $\mathcal{L}^2(\mathbb{Q}^{\gamma,\mathbf{u}^*,\delta_0})$ metric between drifts, thus applying the triangle inequality at the drift level we have that (for simplicitly letting $\gamma = 1$):

$$d(\mathbf{u}^*,\hat{\boldsymbol{v}}) \leq \left(\int_0^1 \mathbb{E}\left[||\boldsymbol{u}_t^* - \boldsymbol{v}_{\sqrt{1-t}}||^2\right] dt\right)^{\frac{1}{2}} + \left(\int_0^1 \mathbb{E}\left||\boldsymbol{v}_{\sqrt{1-t}} - \hat{\boldsymbol{v}}_{\sqrt{1-t}}||^2\right] dt\right)^{\frac{1}{2}}.$$

From [69] we can bound the first term resulting in:

$$d(\mathbf{u}^*,\hat{\boldsymbol{v}}) \leq \epsilon^{1/2} + \left(\int_0^1 \mathbb{E}\left[||\boldsymbol{v}_{\sqrt{1-t}} - \hat{\boldsymbol{v}}_{\sqrt{1-t}}||^2\right] dt\right)^{\frac{1}{2}}$$

Now remembering that the EM drift is given by $\hat{\boldsymbol{v}}_{\sqrt{1-t}}(\boldsymbol{\Theta}_t) = \boldsymbol{v}(\hat{\boldsymbol{\Theta}}_t, \sqrt{1-\Delta t\lceil t/\Delta t\rceil})$, we can use that $\boldsymbol{v}$ is L'-Lipschitz in both arguments, thus:

$$\begin{aligned} d(\mathbf{u}^*, \hat{\boldsymbol{v}}) &\leq \epsilon^{1/2} + \left(L'^2 \int_0^1 \mathbb{E}\left[\left(\left\| \boldsymbol{\Theta}_t - \hat{\boldsymbol{\Theta}}_t \right\| + \Delta t \right)^2 \right] dt \right)^{\frac{1}{2}} \\ &\leq \epsilon^{1/2} + \left(2L'^2 \left(\mathbb{E}\left[\int_0^1 \left\| \boldsymbol{\Theta}_t - \hat{\boldsymbol{\Theta}}_t \right\|^2 dt \right] + \Delta t^2 \right) \right)^{\frac{1}{2}} \\ &\leq \epsilon^{1/2} + \left(2L'^2 \left(\mathbb{E}\left[\max_{0\leq t\leq 1} \left\| \boldsymbol{\Theta}_t - \hat{\boldsymbol{\Theta}}_t \right\|^2 \right] + \Delta t^2 \right) \right)^{\frac{1}{2}}, \end{aligned}$$

which, using the strong convergence of the EM approximation [30], implies:

$$\mathbb{E}\left[\max_{0\leq t\leq 1} \left\| \boldsymbol{\Theta}_t - \hat{\boldsymbol{\Theta}}_t \right\|^2 \right] \leq C_{L'} \Delta t, \tag{A6}$$

thus:

$$d(\mathbf{u}^*, \hat{\boldsymbol{v}}) \leq \epsilon^{1/2} + L'\sqrt{2}\left(\sqrt{C_{L'}\Delta t} + \Delta t \right).$$

Squaring both sides and applying the data processing inequality completes the proof. □

Appendix B Connections to VI

We first start by making the connection in a simpler case – when the prior of our Bayesian model is given by a Gaussian distribution with variance γ, that is $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\mathbf{0}, \gamma\mathbb{I}_d)$.

Observation 1 *When $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\mathbf{0}, \gamma\mathbb{I}_d)$, it follows that the N-SFP objective in Equation 8 corresponds to the negative ELBO of the model:*

$$\begin{aligned} d\boldsymbol{\Theta}_t &= \sqrt{\gamma} d\mathbf{B}_t, \quad \boldsymbol{\Theta}_0 \sim \delta_0, \\ \mathbf{x}_i &\sim p(\mathbf{x}_i|\boldsymbol{\Theta}_1). \end{aligned} \tag{B7}$$

Proof Substituting $p(\boldsymbol{\theta})$ into Equation 8 yields

$$\mathbf{u}^* = \arg\min_{\mathbf{u}\in\mathcal{U}} \mathbb{E}_{\boldsymbol{\Theta}\sim\mathbb{Q}^{0,\delta_0}} \left[\frac{1}{2\gamma} \int_0^1 \|\mathbf{u}(t, \boldsymbol{\Theta}_t)\|^2 dt - \ln p(\boldsymbol{X}|\boldsymbol{\Theta}_1) \right]. \tag{B8}$$

Then, from [8, 68, 69] we know that the term $\mathbb{E}\left[\int_0^1 \|\boldsymbol{u}_t\|^2 dt - \ln p(\boldsymbol{X}|\boldsymbol{\Theta}_1) \right]$ is the negative ELBO of the model specified in Equation B7. □

While the above observation highlights a specific connection between N-SFP and traditional VBI (Variational Bayesian Inference), it is limited to Bayesian models that are specified with Gaussian priors. To extend the result, we take inspiration from the recursive nature of Bayesian updates in the following result.

Lemma 1 *The SBP $\inf_{\mathbb{Q}\in\mathcal{D}(\delta_0,\, p(\boldsymbol{\theta}|\boldsymbol{X}))} D_{\mathrm{KL}}\left(\mathbb{Q}||\mathbb{S}\right)$ with reference process $\mathbb{S}$ described by*

$$\boldsymbol{\Theta}_0 \sim \delta_0 \tag{B9}$$

$$d\boldsymbol{\Theta}_t = \nabla \ln Q^{\gamma}_{1-t}\left[\frac{p(\boldsymbol{\Theta}_t)}{\mathcal{N}(\boldsymbol{\Theta}_t|\mathbf{0}, \gamma\mathbb{I}_d)} \right] + \sqrt{\gamma} d\mathbf{B}_t, \tag{B10}$$

corresponds to maximising the ELBO of the model:

$$\boldsymbol{\Theta}_0 \sim \delta_0,$$

$$d\boldsymbol{\Theta}_t = \nabla \ln Q_{1-t}^{\gamma}\left[\frac{p(\boldsymbol{\Theta}_t)}{\mathcal{N}(\boldsymbol{\Theta}_t|\mathbf{0},\gamma\mathbb{I}_d)}\right] + \sqrt{\gamma}d\mathbf{B}_t,,$$
$$\mathbf{x}_i \sim p(\mathbf{x}_i|\boldsymbol{\Theta}_1). \tag{B11}$$

Proof For brevity let $\mathbf{u}^0(t,\boldsymbol{\theta}) = \nabla \ln Q_{1-t}^{\gamma}\left[\frac{p(\boldsymbol{\theta})}{\mathcal{N}(\boldsymbol{\theta}|\mathbf{0},\gamma\mathbb{I}_d)}\right]$. First notice that the time-one marginals of $\mathbb{S}$ are given by the Bayesian prior:

$$(\boldsymbol{\Theta}_1)_{\#}\mathbb{S} = p(\boldsymbol{\theta})d\boldsymbol{\theta}$$

Now from [49, 58] we know that the Schrödinger system is given by:

$$\phi_0(\boldsymbol{\theta}_0)\int p(\boldsymbol{\theta}_0,0,\boldsymbol{\theta}_1,1)\hat{\phi}_1(\boldsymbol{\theta}_1)d\boldsymbol{\theta}_1 = \delta_0(\boldsymbol{\theta}_0), \tag{B12}$$

$$\hat{\phi}_1(\boldsymbol{\theta}_1)\int p(\boldsymbol{\theta}_0,0,\boldsymbol{\theta}_1,1)\phi_0(\boldsymbol{\theta}_0)d\boldsymbol{\theta}_0 = p(\boldsymbol{\theta}_1|\boldsymbol{X}), \tag{B13}$$

where Equation B12 can be given a rigorous meaning in weak form (that is, by integrating against suitable test functions). Notice $\phi_0 = \delta_0$ and thus it follows that

$$\hat{\phi}_1(\boldsymbol{\theta}) = \frac{p(\boldsymbol{\theta}|\boldsymbol{X})}{p(0,0,\boldsymbol{\theta},1)} = \frac{p(\boldsymbol{\theta}|\boldsymbol{X})}{p(\boldsymbol{\theta})} = \frac{p(\boldsymbol{X}|\boldsymbol{\theta})}{\mathcal{Z}}. \tag{B14}$$

By [13, 57, 58] the optimal drift is given by:

$$\mathbf{u}^*(t,\boldsymbol{\theta}) = \gamma\nabla \ln \mathbb{E}[p(\boldsymbol{X}|\boldsymbol{\Theta}_1)|\boldsymbol{\Theta}_t = \boldsymbol{\theta}], \tag{B15}$$

where the expectation is taken with respect to the reference process $\mathbb{S}$. Now if we let $v(\boldsymbol{\theta},t) = -\ln \mathbb{E}[p(\boldsymbol{X}|\boldsymbol{\Theta}_1)|\boldsymbol{\Theta}_t = \boldsymbol{\theta}]$ be our value function then via the linearisation of the Hamilton-Bellman-Jacobi Equation through Fleming's logarithmic transform [41, 67, 69] it follows that said value function satisfies:

$$v(\boldsymbol{\theta},t) = \min_{\mathbf{u}\in\mathcal{U}} \mathbb{E}\left[\frac{1}{2\gamma}\int_t^1 \left\|\mathbf{u}(t,\boldsymbol{\Theta}_t) - \mathbf{u}^0(t,\boldsymbol{\Theta}_t)\right\|^2 dt - \ln p(\boldsymbol{X}|\boldsymbol{\Theta}_1)\middle|\boldsymbol{\Theta}_t = \boldsymbol{\theta}\right], \tag{B16}$$

and thus $\mathbf{u}^*(t,\boldsymbol{\theta}) = \gamma\nabla \ln \mathbb{E}[p(\boldsymbol{X}|\boldsymbol{\Theta}_1)|\boldsymbol{\Theta}_t = \boldsymbol{\theta}]$ is a minimiser to:

$$\mathbf{u}^* = \arg\min_{\mathbf{u}\in\mathcal{U}} \mathbb{E}\left[\frac{1}{2\gamma}\int_0^1 \left\|\mathbf{u}(t,\boldsymbol{\Theta}_t) - \mathbf{u}^0(t,\boldsymbol{\Theta}_t)\right\|^2 dt - \ln p(\boldsymbol{X}|\boldsymbol{\Theta}_1)\right]. \tag{B17}$$

□

Appendix C Stochastic Variational Inference

For a Bayesian model having the structure specified by equation 1 the objective in equation 8 can be written as follows:

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\Theta}\sim\mathbb{Q}^{\mathbf{u},\delta_0}}&\left[\frac{1}{2\gamma}\int_0^1 \|\mathbf{u}(t,\boldsymbol{\Theta}_t)\|^2 dt - \ln\frac{p(\boldsymbol{X}|\boldsymbol{\Theta}_1)p(\boldsymbol{\Theta}_1)}{\mathcal{N}(\boldsymbol{\Theta}_1|\mathbf{0},\gamma\mathbb{I}_d)}\right]\\ &= \mathbb{E}\left[\frac{1}{2\gamma}\int_0^1 \|\mathbf{u}(t,\boldsymbol{\Theta}_t)\|^2 dt - \ln\frac{p(\boldsymbol{\Theta}_1)}{\mathcal{N}(\boldsymbol{\Theta}_1|\mathbf{0},\gamma\mathbb{I}_d)}\right] + \sum_{i=1}^N \mathbb{E}\left[\ln p(\mathbf{x}_i|\boldsymbol{\Theta}_1)\right],\end{aligned} \tag{C18}$$

where the last term can be written as:

$$\sum_{i=1}^{N} \mathbb{E}\left[\ln p(\mathbf{x}_i|\boldsymbol{\Theta}_1)\right] = \frac{N}{B}\mathbb{E}_{\mathbf{x}_{k_i}\sim\mathcal{D}}\left[\sum_{i=1}^{B} \mathbb{E}\left[\ln p(\mathbf{x}_{k_i}|\boldsymbol{\Theta}_1)\right]\right] \tag{C19}$$

That is, it is possible to obtain an unbiased estimate of the objective (and its gradients) by subsampling the data with random batches of size B and using the scaling $\frac{N}{B}$. A version of the algorithm with Euler-Maruyama discretization of the SDE is given in Algorithm 1.

Appendix D Decoupled Drift Results

First let us consider the setting where the local variables are fully independent, that is, $\boldsymbol{\theta}_i \perp\!\!\!\perp \boldsymbol{\theta}_j$.

Remark 4 *The heat semigroup preserves fully factored (mean-field) distributions thus the Föllmer drift is decoupled.*

In this setting we can parametrise the dimensions of the drift which correspond to local variables in a decoupled manner, $[\mathbf{u}_t]_{\boldsymbol{\theta}_i} = u^{\boldsymbol{\theta}_i}(t, \boldsymbol{\theta}_i, \boldsymbol{x}_i)$. This amortised parametrisation [44] allows us to carry out gradient estimates using a mini-batch [37] rather than hold the whole state space in memory.

Remark 2 *The heat semigroup does not preserve conditional independence structure in the drift. That is, the optimal drift does not decouple and as a result depends on the full state space.*

Proof Consider the following distribution:

$$\mathcal{N}(x|z,0)\mathcal{N}(y|z,0)\mathcal{N}(z|0,1) \tag{D20}$$

We want to estimate:

$$\mathbb{E}\left[\frac{\mathcal{N}(X+x|Z+z,1)\mathcal{N}(Y+y|Z+z,1)\mathcal{N}(Z+z|1,0)}{\mathcal{N}(X+x|0,1)\mathcal{N}(Y+y|0,1)\mathcal{N}(Z+z|0,1)}\right], \tag{D21}$$

where $X, Y, Z \sim \mathcal{N}(0, \sqrt{1-t})$. From

$$\mathbb{E}\left[\frac{\mathcal{N}(X+x|Z+z,1)\mathcal{N}(Y+y|Z+z,1)}{\mathcal{N}(X+x|0,1)\mathcal{N}(Y+y|0,1)}\right] \tag{D22}$$

we can easily see that the above no longer has conditional independence structure and thus when taking its logarithmic derivative the drift does not decouple. □

Remark 3 *An SDE parametrised with a decoupled drift $[\mathbf{u}_t]_{\boldsymbol{\theta}_i} = u(t, \boldsymbol{\theta}_i, \Phi, \boldsymbol{x}_i)$ can reach transition densities which do not factor.*

Proof Consider the linear time-homogeneous SDE:

$$d\boldsymbol{\Theta}_t = \boldsymbol{A}\boldsymbol{\Theta}_t dt + \gamma d\boldsymbol{W}_t, \quad \boldsymbol{\Theta}_0 = 0, \tag{D23}$$

where:

$$[\boldsymbol{A}]_{ij} = \delta_{ij} + i\delta_{1j}, \tag{D24}$$

then this SDE admits a closed form solution:

$$\boldsymbol{\Theta}_t = \gamma \int_0^t \exp\left(\boldsymbol{A}(t-s)\right) d\boldsymbol{W}_s, \tag{D25}$$

which is a Gauss-Markov process with 0 mean and covariance matrix:

$$\boldsymbol{\Sigma}(t) = \gamma^2 \int_0^t \exp\left(\boldsymbol{A}(t-s)\right) \exp\left(\boldsymbol{A}(t-s)\right)^\top ds \tag{D26}$$

We can carry out the matrix exponential through the eigendecomposition of $\boldsymbol{A}$, for simplicity let us consider the 3-dimensional case:

$$\exp\left(\boldsymbol{A}(t-s)\right) = Se^{D(t-s)}S^{-1} = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 2 \\ 0 & 0 & 2 \end{pmatrix} \begin{pmatrix} e^{t-s} & 0 & 0 \\ 0 & e^{t-s} & 0 \\ 0 & 0 & e^{3(t-s)} \end{pmatrix} \begin{pmatrix} 0 & 1 & -1 \\ 1 & 0 & -1/2 \\ 0 & 0 & 1/2 \end{pmatrix} \tag{D27}$$

From this we see that:

$$\begin{aligned} \exp\left(\boldsymbol{A}(t-s)\right) \exp\left(\boldsymbol{A}(t-s)\right)^\top &= Se^{D(t-s)}S^{-1}(Se^{D(t-s)}S^{-1})^\top && \text{(D28)} \\ &= Se^{D(t-s)}S^{-1}S^{-\top}e^{D(t-s)}S^\top && \text{(D29)} \\ &= \frac{1}{4}Se^{D(t-s)} \begin{pmatrix} 8 & 2 & -2 \\ 2 & 5 & -1 \\ -2 & -1 & 1 \end{pmatrix} e^{D(t-s)}S^\top && \text{(D30)} \\ &= \frac{1}{4}S \begin{pmatrix} 8e^{2(t-s)} & 2e^{2(t-s)} & -2e^{4(t-s)} \\ 2e^{2(t-s)} & 5e^{2(t-s)} & -e^{4(t-s)} \\ -2e^{4(t-s)} & -e^{4(t-s)} & e^{6(t-s)} \end{pmatrix} S^\top && \text{(D31)} \end{aligned}$$

Integrating wrt to s yields:

$$\begin{aligned} \int \exp\left(\boldsymbol{A}(t-s)\right) \exp\left(\boldsymbol{A}(t-s)\right)^\top ds &= \frac{1}{4}S \begin{pmatrix} 4 & 1 & -\frac{1}{2} \\ 1 & \frac{5}{2} & -\frac{1}{4} \\ -\frac{1}{2} & -\frac{1}{4} & \frac{1}{6} \end{pmatrix} S^\top && \text{(D32)} \\ &= \frac{1}{24} \begin{pmatrix} 13 & 2 & -1 \\ 2 & 16 & -2 \\ -1 & -2 & 4 \end{pmatrix}. && \text{(D33)} \end{aligned}$$

The covariance matrix is dense at all times and thus the density $\text{Law}(\boldsymbol{\Theta}_t) = \mathcal{N}(\boldsymbol{\mu}(t), \boldsymbol{\Sigma}(t))$ does not factor (is a fully joint distribution). This example motivates that even with the decoupled drift we can reach coupled distributions.

□

Appendix E Low Variance Estimators and Sticking the Landing

Theorem 1 *The STL estimator proposed in [74] satisfies*

$$\frac{\mathrm{d}}{\mathrm{d}\varepsilon}\mathcal{F}(\mathbf{u}^* + \varepsilon\boldsymbol{\phi})\Big|_{\varepsilon=0} = 0, \tag{E34}$$

almost surely, for all smooth and bounded perturbations $\boldsymbol{\phi}$.

Proof Let us decompose $\mathcal{F}$ in the following way:

$$\mathcal{F}(\mathbf{u}) = \mathcal{F}_0(\mathbf{u}) + \mathcal{F}_1(\mathbf{u}) \tag{E35}$$

where (denoting the terminal cost with g):

$$\mathcal{F}_0(\mathbf{u}) = \frac{1}{2\gamma}\int_0^1 \|\mathbf{u}(t,\boldsymbol{\Theta}_t)\|^2\, dt + g(\boldsymbol{\Theta}_1) \tag{E36}$$

$$\mathcal{F}_1(\mathbf{u}) = \frac{1}{\sqrt{\gamma}}\int_0^1 \mathbf{u}^{\perp}(t,\boldsymbol{\Theta}_t)^{\top} d\mathbf{B}_t \tag{E37}$$

Denoting $\boldsymbol{\Theta}^{\mathbf{u}} \sim \mathbb{Q}^{\mathbf{u},\delta_0}$, from [54], Theorem 5.3.1, Equation 133 it follows that:

$$\frac{\mathrm{d}}{\mathrm{d}\varepsilon}\mathcal{F}_0(\mathbf{u}^* + \varepsilon\boldsymbol{\phi})\Big|_{\varepsilon=0} = -\frac{1}{\sqrt{\gamma}}\int_0^1 \boldsymbol{A}_t \cdot (\nabla \mathbf{u}_t^*)(\boldsymbol{\Theta}_t^{\mathbf{u}^*})\, \mathrm{d}\mathbf{B}_t, \tag{E38}$$

almost surely, where $\boldsymbol{A}_t$ is defined as

$$\boldsymbol{A}_t^{\phi} = \frac{\mathrm{d}\boldsymbol{\Theta}_t^{\mathbf{u}^*+\varepsilon\boldsymbol{\phi}}}{\mathrm{d}\varepsilon}\Big|_{\varepsilon=0} \tag{E39}$$

and satisfies:

$$\mathrm{d}\boldsymbol{A}_t^{\phi} = \phi_t(\boldsymbol{\Theta}_t^{\mathbf{u}^*})\,\mathrm{d}t + (\nabla \mathbf{u}^*)^{\top}(\boldsymbol{\Theta}_t^{\mathbf{u}^*})\boldsymbol{A}_t^{\phi}\,\mathrm{d}t, \qquad \boldsymbol{A}_0^{\phi} = 0. \tag{E40}$$

Similarly via the chain rule it follows that:

$$\frac{\mathrm{d}}{\mathrm{d}\varepsilon}\mathcal{F}_1(\mathbf{u}^* + \varepsilon\boldsymbol{\phi})\Big|_{\varepsilon=0} = \frac{\mathrm{d}}{\mathrm{d}\varepsilon}\left(\frac{1}{\sqrt{\gamma}}\int_0^1 \mathbf{u}_t^*(\boldsymbol{\Theta}_t^{\mathbf{u}^*+\varepsilon\boldsymbol{\phi}})^{\top}\mathrm{d}\mathbf{B}_t\right)\Big|_{\varepsilon=0} = \frac{1}{\sqrt{\gamma}}\int_0^1 \boldsymbol{A}_t^{\phi}\cdot(\nabla \mathbf{u}_t^*)(\boldsymbol{\Theta}_t^{\mathbf{u}^*})\mathrm{d}\mathbf{B}_t \tag{E41}$$

almost surely, combining these results we can see that $\frac{\mathrm{d}}{\mathrm{d}\varepsilon}\mathcal{F}(\mathbf{u}^* + \varepsilon\boldsymbol{\phi})\Big|_{\varepsilon=0} = 0$ almost surely as required. □

Appendix F Stabilising MC-SFS Implementation

We found the estimators proposed in [39] (Equations 2.20 or 2.21, and Algorithm 2 in [39]) to be very numerically unstable. Even in two dimensions the montecarlo estimator of the drift evaluated to nans and infs on more than 50% of the generated samples. This is due to the RND f of Equation 7 often evaluating to either 0 due to underflow or a very small number resulting in Equation 7 becoming very large and unstable.

In order to alleviate this we propose the a novel modified logsmexp reformulation of Equation 7:

Lemma 2 *(Stable MC-SFS) The MC-SFS estimator*

$$\hat{\mathbf{u}}^*(t,\boldsymbol{x}) = \frac{\mathbb{E}_{\boldsymbol{z}\sim\hat{P}}[\boldsymbol{z}_s f(\boldsymbol{x}+\sqrt{1-t}\boldsymbol{z})]}{\mathbb{E}_{\boldsymbol{z}\sim\hat{P}}[\sqrt{1-t} f(\boldsymbol{x}+\sqrt{1-t}\boldsymbol{z})]}, \tag{F42}$$

Where $\hat{P}$ is the empirical measure:

$$\hat{P} = \frac{1}{S}\sum_{s=1}^{S}\delta_{\boldsymbol{z}_s} \tag{F43}$$

Can be re-expresssed as:

$$\hat{\mathbf{u}}^*(t,\boldsymbol{x}) = \exp\left(\operatorname*{logsumexp}_s g_{\boldsymbol{x}}^{+}(\boldsymbol{z}_s) - \operatorname*{logsumexp}_s \ln \mathcal{Z}_s\right) \tag{F44}$$

$$- \exp\left(\operatorname*{logsumexp}_s g_{\boldsymbol{x}}^{-}(\boldsymbol{z}) - \operatorname*{logsumexp}_s \ln \mathcal{Z}_s\right) \tag{F45}$$

where:

$$g^{+}_{\boldsymbol{x}}(\boldsymbol{z}_s) = \begin{cases} \ln \boldsymbol{z}_s f(\boldsymbol{x} + \sqrt{1-t}\boldsymbol{z}_s) & \text{if } \boldsymbol{z}_s > 0 \\ 0 & \text{otherwise} \end{cases} \tag{F46}$$

$$g^{-}_{\boldsymbol{x}}(\boldsymbol{z}_s) = \begin{cases} \ln \boldsymbol{z}_s f(\boldsymbol{x} + \sqrt{1-t}\boldsymbol{z}_s) & \text{if } \boldsymbol{z}_s < 0 \\ 0 & \text{otherwise} \end{cases} \tag{F47}$$

and $\ln \mathcal{Z}_s = \ln \sqrt{1-t} + \ln f(\boldsymbol{x} + \sqrt{1-t}\boldsymbol{z}_s)$

Proof Firstly notice that the logsumexp formula cannot be applied to the numerator as the terms $\boldsymbol{z}_s f(\boldsymbol{x} + \sqrt{1-t}\boldsymbol{z}_s)$ in the numerator can take on negative values and thus we cannot take the log.

In order to take log the note that $\mathbb{E}_{\hat{P}}[f]$ is a Lebesgue–Stieltjes integral and thus by construction we can decompose it into positive and negative parts:

$$\hat{\mathbf{u}}^{*}_{t}(\boldsymbol{x}) = \frac{\mathbb{E}_{\boldsymbol{z}\sim\hat{P}}[(\boldsymbol{z}_s f(\boldsymbol{x} + \sqrt{1-t}\boldsymbol{z}))]}{\mathbb{E}_{\boldsymbol{z}\sim\hat{P}}[\sqrt{1-t} f(\boldsymbol{x} + \sqrt{1-t}\boldsymbol{z})]} = \frac{\mathbb{E}_{\boldsymbol{z}\sim\hat{P}}[(\boldsymbol{z}_s f(\boldsymbol{x} + \sqrt{1-t}\boldsymbol{z}))^{+}]}{\mathbb{E}_{\boldsymbol{z}\sim\hat{P}}[\sqrt{1-t} f(\boldsymbol{x} + \sqrt{1-t}\boldsymbol{z})]} - \frac{\mathbb{E}_{\boldsymbol{z}\sim\hat{P}}[(\boldsymbol{z}_s f(\boldsymbol{x} + \sqrt{1-t}\boldsymbol{z}))^{-}]}{\mathbb{E}_{\boldsymbol{z}\sim\hat{P}}[\sqrt{1-t} f(\boldsymbol{x} + \sqrt{1-t}\boldsymbol{z})]} \tag{F48}$$

wlog consider the first term:

$$\frac{\mathbb{E}_{\boldsymbol{z}\sim\hat{P}}[(\boldsymbol{z}_s f(\boldsymbol{x} + \sqrt{1-t}\boldsymbol{z}))^{+}]}{\mathbb{E}_{\boldsymbol{z}\sim\hat{P}}[\sqrt{1-t} f(\boldsymbol{x} + \sqrt{1-t}\boldsymbol{z})]} = \exp\left(\ln \mathbb{E}_{\boldsymbol{z}\sim\hat{P}}[(\boldsymbol{z}_s f(\boldsymbol{x} + \sqrt{1-t}\boldsymbol{z}))^{+}] - \ln E_{\boldsymbol{z}\sim\hat{P}}[\sqrt{1-t} f(\boldsymbol{x} + \sqrt{1-t}\boldsymbol{z})]\right) \tag{F49}$$

and similarly for the second, at this point we can trivially apply the log sum exp formula to each of the exponents separately as their integrands are positive. □

For efficient implementation we first separate the samples into positive and negative and then proceed to compute each of the g^{+} and g^{-} terms separately which avoids evaluating any $\ln 0$ terms. We found this formula to have no numerical instabilities in our experiments ranging up to high dimensional cases $d = 2^{12}$ without issue.

Appendix G Sensitivity of hyperparameters to Hypespectral Unmixing Results

While we were able to find step size schedules for SGLD that would work well for the Hyperspectral image data, it is important to note that it was due to heavy tuning and a stroke of luck. As shown in H1 there are four parameters to adjust for the step size scheduling of SGLD and the resulting performance is very sensitive to all of them. To illustrate this, we fixed the parameters associated to σ^2 as given in H1, and varied the others. The resulting samples are provided in figure G1.

In contrast, N-SFS has only one tunable parameter, which impacts the results much less, as shown in figures G2 and G3.

Appendix H Experimental Details and Further Results

H.1 Method Hyperparameters

In Table H1 we show the experimental configuration of the trialled algorithms across all datasets. For the selected values of γ we ran a small grid search $\gamma \in \{0.5^2, 0.2^2, 0.1^2, 0.05^2, 0.01^2\}$ and selected the γ with best training set results.

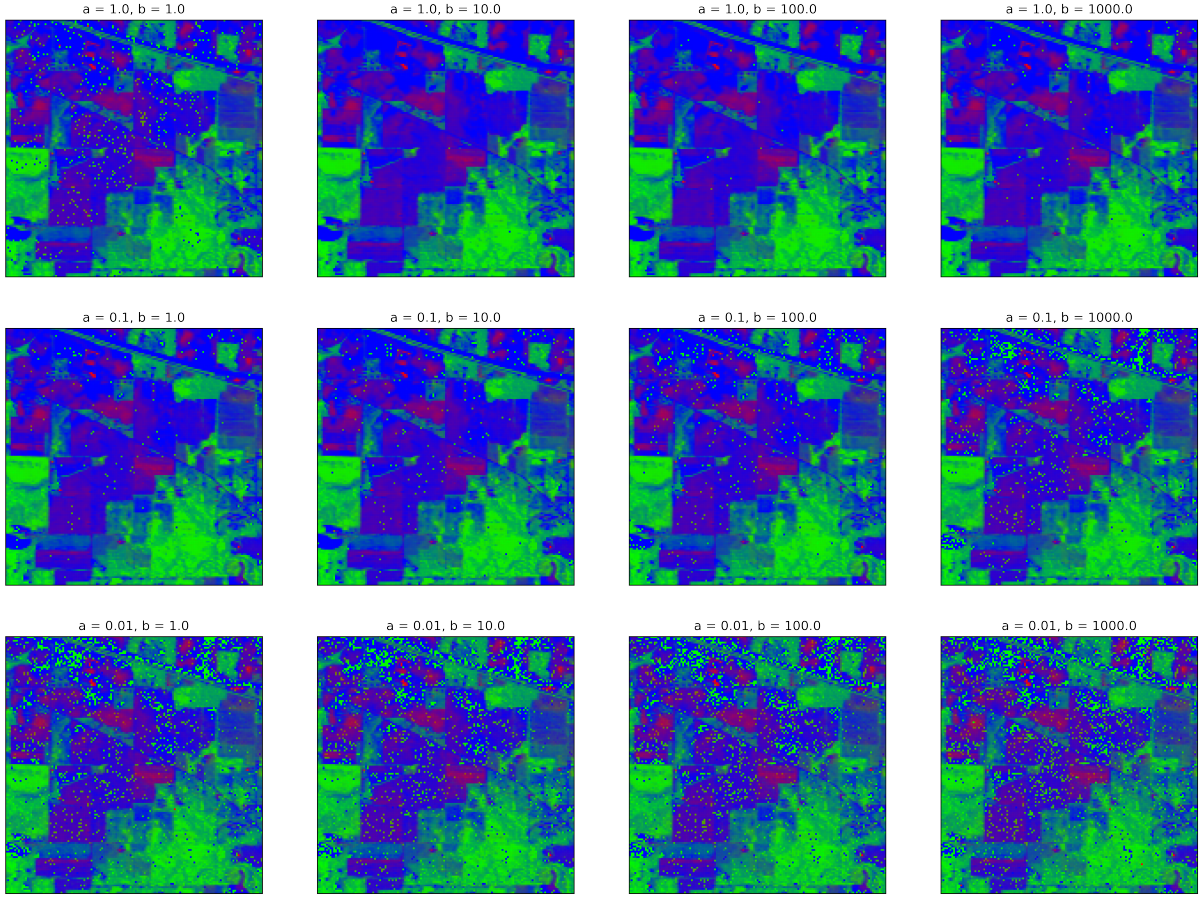

Fig. G1 SGLD sensitivity to step size scheduling

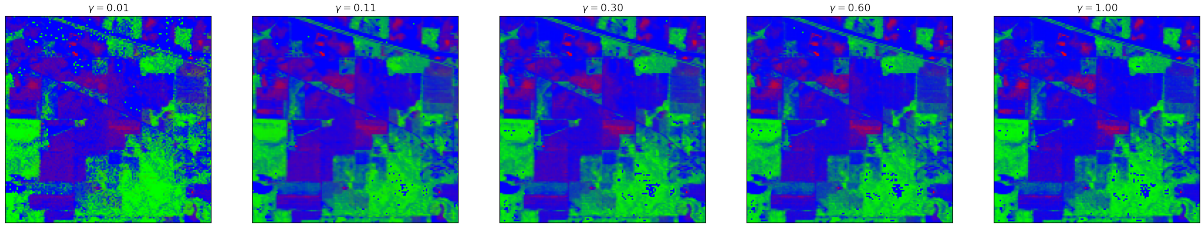

Fig. G2 N-SFS sensitivity to γ

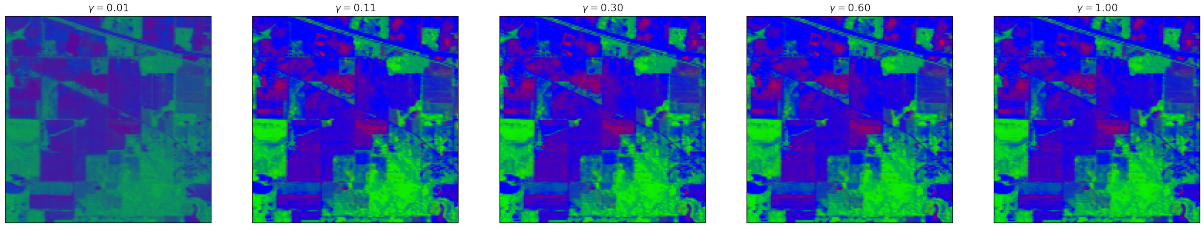

Fig. G3 Decoupled N-SFS sensitivity to γ

H.2 Step Function Dataset

Here we describe in detail how the step function dataset was generated:

$$y(x) = \mathbb{1}_{x \geq 0} + \epsilon, \quad \epsilon \sim \mathcal{N}(0, 0.1) \tag{H50}$$

Where:

- $\sigma_y = 0.1$
- $N_{\text{train}} = 100$, $N_{\text{test}} = 100$
- $x_{\text{train}} \in (-3.5, 3.5)$
- $x_{\text{test}} \in (-10, 10)$

Table H1 Hyper parameter configuration for methods and optimisers.

Method	Hyperparameters	Experiments					
		Step Function	MNIST	CIFAR10	Hyperspectral Unmixing	LogReg	ICA
N-SFS	Optimser	Adam	Adam	Adam	Adam / Adam	Adam	Adam
	Optimiser step size	10^{-4}	10^{-5}	10^{-5}	10^{-5}	10^{-4}	10^{-4}
	$\boldsymbol{\Theta}$ batch size	32	32	32	32	32	32
	Data batch size	32	50	50	Whole dataset	Whole train set	10
	# of iterations	300	18750	18750	2000	300	2832
	# of posterior samples	100	100	100	20	100	100
	γ	0.05^2	0.1^2	0.05^2	0.2^2	0.2^2	0.01^2
	EM train Δt_{train}	0.05	0.05	0.05	0.05	0.05	0.05
	EM test Δt_{test}	0.01	0.01	0.01	0.01	0.01	0.01
SGLD	Adaptive step schedule	$\lambda(i) = \frac{a}{(i+b)^\gamma}$	$\lambda(i) = \frac{a}{(i+b)^\gamma}$	$\lambda(i) = \frac{a}{(i+b)^\gamma}$	$\lambda_{\sigma^2}(i) = \frac{a_{\sigma^2}}{(i+b_{\sigma^2})^\gamma}, \lambda_A(i) = \frac{a_A}{(i+b_A)^\gamma}$	$\lambda(i) = \frac{a}{(i+b)^\gamma}$	$\lambda(i) = \frac{a}{(i+b)^\gamma}$
	a	10^{-3}	7×10^{-5}	10^{-4}	$a_A = 1.0, a_{\sigma^2} = 10^{-6}$	10^{-4}	10^{-4}
	b	10	1	1	$b_A = 10.0, b_{\sigma^2} = 1.0$	1	1
	γ	0.55	0.55	0.55	0.55	0.55	0.55
	Posterior Samples	100	100	100	20	100	100
	Data batch size	32	32	32	Whole dataset	32	32
	# of iterations	300	18750	18750	10000	300	2832
SGD	step size	10^{-2}	10^{-1}	10^{-3}	-	-	-
	Data batch size	32	32	32	-	-	-
	# of iterations	300	18750	18750	-	-	-

H.3 Föllmer Drift Architecture

Across all experiments (with the exception of the MNIST dataset) we used the same architecture to parametrise the Föllmer drift:

```
class SimpleForwardNetBN(torch.nn.Module):

    def __init__(self, input_dim=1, width=20):
        super(SimpleForwardNetBN, self).__init__()

        self.input_dim = input_dim

        self.nn = torch.nn.Sequential(
            torch.nn.Linear(input_dim + 1, width),
            torch.nn.BatchNorm1d(width, affine=False),
            torch.nn.Softplus(),
            torch.nn.Linear(width, width),
            torch.nn.BatchNorm1d(width, affine=False),
            torch.nn.Softplus(),
            torch.nn.Linear(width, width),
            torch.nn.BatchNorm1d(width, affine=False),
            torch.nn.Softplus(),
            torch.nn.Linear(width, width),
            torch.nn.BatchNorm1d(width, affine=False),
            torch.nn.Softplus(),
            torch.nn.Linear(width, input_dim)
        )

        self.nn[-1].weight.data.fill_(0.0)
        self.nn[-1].bias.data.fill_(0.0)
```

Listing 1 Simple architecture for drift.

Note the weights and biases of the final layer are initialised to 0 in order to start the process at a Brownian motion matching the SBP prior.

For the MNIST dataset we used the score network proposed in [11]. We aimed in using this same architecture for the CIFAR10 experiments however we were unable to train it stably.

For Hyperspectral Unmixing dataset we used this architecture for N-SFS with full drift, but had to devise a different architecture for decoupled drifts, as shown below.

```
class ResNetScoreNetwork(torch.nn.Module):

    def __init__(self, input_dim: int, final_zero=False):
        super().__init__()
        res_block_initial_widths = [300, 300, 300]
        res_block_final_widths = [300, 300, 300]
        res_block_inner_layers = [300, 300, 300]

        self.input_dim = input_dim

        self.temb_dim = 128

        # ResBlock Sequence
        res_layers = []
        initial_dim = input_dim
        for initial, final in zip(res_block_initial_widths, res_block_final_widths):
            res_layers.append(ResBlock(initial_dim, initial, final, res_block_inner_layers, torch.nn.
    Softplus()))
            initial_dim = initial + final
        self.res_sequence = torch.nn.Sequential(*res_layers)

        # Time FCBlock
        self.time_block = torch.nn.Sequential(torch.nn.Linear(self.temb_dim, self.temb_dim * 2), torch
    .nn.Softplus())

        # Final_block
        self.final_block = torch.nn.Sequential(torch.nn.Linear(self.temb_dim * 2 + initial_dim,
    input_dim))
```

Listing 2 Score Network architecture for drift.

```
class DecoupledDrift(AbstractDrift):

```

```
    def __init__(self, global_dim=1, local_dim=1, data_dim=1, width=20):
        super(DecoupledDrift, self).__init__()

        self.global_dim = global_dim
        self.local_dim = local_dim
        self.data_dim = data_dim

        self.nn = torch.nn.Sequential(
            torch.nn.Linear(global_dim + local_dim + data_dim + 1, width), torch.nn.BatchNorm1d(width,
     affine=False), torch.nn.Softplus(),
            torch.nn.Linear(width, width), torch.nn.BatchNorm1d(width, affine=False), torch.nn.
    Softplus(),
            torch.nn.Linear(width, width), torch.nn.BatchNorm1d(width, affine=False), torch.nn.
    Softplus(),
            torch.nn.Linear(width, width), torch.nn.BatchNorm1d(width, affine=False), torch.nn.
    Softplus(),
            torch.nn.Linear(width, local_dim)
        )

        self.nn[-1].weight.data.fill_(0.0)
        self.nn[-1].bias.data.fill_(0.0)
```

Listing 3 Decoupled Drift network for local parameters

H.4 BNN Architectures

For the step function dataset we used the following architecture:

```
class DNN_StepFunction(torch.nn.Module):

    def __init__(self, input_dim=1, output_dim=1):
        super(DNN, self).__init__()

        self.output_dim = output_dim
        self.input_dim = input_dim

        self.nn = torch.nn.Sequential(
            torch.nn.Linear(input_dim, 100),
            torch.nn.ReLU(),
            torch.nn.Linear(100, 100),
            torch.nn.ReLU(),
            torch.nn.Linear(100, output_dim)
        )
```

Listing 4 Architecture for step function dataset.

For LeNet5 the architecture used was:

```
class LeNet5(torch.nn.Module):

    def __init__(self, n_classes):
        super(LeNet5, self).__init__()

        self.feature_extractor = torch.nn.Sequential(
            torch.nn.Conv2d(
                in_channels=1, out_channels=6,
                kernel_size=5, stride=1
            ),
            torch.nn.Tanh(),
            torch.nn.AvgPool2d(kernel_size=2),
            torch.nn.Conv2d(
                in_channels=6, out_channels=16,
                kernel_size=5, stride=1
            ),
            torch.nn.Tanh(),
            torch.nn.AvgPool2d(kernel_size=2),
        )

        self.classifier = torch.nn.Sequential(
            torch.nn.Linear(in_features=256, out_features=120),
            torch.nn.Tanh(),
            torch.nn.Linear(in_features=120, out_features=84),
            torch.nn.Tanh(),
            torch.nn.Linear(in_features=84, out_features=n_classes),
```

```
27          )
```

Listing 5 Architecture for MNIST.

The same layer structure as in LeNet5 was used for the CIFAR10 dataset,and with a difference in the number of channels and size of filters. Exact details can be found in the code repository.

H.5 Likelihood and Prior Hyperparameters

In Table H.5 we describe the hyperparameters of each Bayesian model as well as their priors and likelihood.

Model	Hyperparameters	Values
Step Function	Prior	$\mathcal{N}(\mathbf{0}, \sigma_\theta^2 \mathbb{I})$
	Likelihood	$\mathcal{N}(\boldsymbol{y}_i \vert f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \sigma_y^2 \mathbb{I})$
	σ_θ	1
	σ_y	0.1
MNIST	Prior	$\mathcal{N}(\mathbf{0}, \sigma_\theta^2 \mathbb{I})$
	Likelihood	$\text{Cat}(f_\theta(\boldsymbol{x}_i))$
	σ_θ	1
CIFAR10	Prior	$\mathcal{N}(\mathbf{0}, \sigma_\theta^2 \mathbb{I})$
	Likelihood	$\text{Cat}(f_\theta(\boldsymbol{x}_i))$
	σ_θ	1
Hyperspectral Unmixing	Prior	$p(\sigma^2) = \mathbf{1}_{[0,1]}(\sigma^2), p(a_p) = \mathbf{1}_{\Delta_R}(\boldsymbol{a}_p)$
	Likelihood	$\mathcal{N}(\boldsymbol{M}\boldsymbol{a}_p;\ \vert\vert\boldsymbol{a}_p\vert\vert^2 \sigma^2 I)$
Log Reg	Prior	$\text{Laplace}(\mathbf{0}, \sigma_\theta,)$
	Likelihood	$\text{Bern}(\text{Sigmoid}_\theta)$
	σ_θ	1
ICA	Prior	$\mathcal{N}(\mathbf{0}, \sigma_\theta^2 \mathbb{I})$
	Likelihood	$\prod_i \frac{1}{4\cosh^2(\frac{\boldsymbol{\theta}_i^\top \boldsymbol{x}}{2})}$
	σ_θ	1

Table H2 Specification of Bayesian models.

References

[1] Amari, S.-i., Cichocki, A., Yang, H. H., et al. (1996). A new learning algorithm for blind signal separation. In *Advances in neural information processing systems*, pages 757–763. Morgan Kaufmann Publishers.

[2] Bartholomew-Biggs, M., Brown, S., Christianson, B., and Dixon, L. (2000). Automatic differentiation of algorithms. *Journal of Computational and Applied Mathematics*, 124(1-2):171–190.

[3] Belkin, M., Hsu, D., Ma, S., and Mandal, S. (2019). Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854.

[4] Bernton, E., Heng, J., Doucet, A., and Jacob, P. E. (2019). Schrödinger bridge samplers. *arXiv preprint*.

[5] Bioucas-Dias, J. M., Plaza, A., Dobigeon, N., Parente, M., Du, Q., Gader, P., and Chanussot, J. (2012). Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 5(2):354–379.

[6] Bishop, C. M. (1999). Bayesian PCA. *Advances in neural information processing systems*, pages 382–388.

[7] Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

[8] Boué, M. and Dupuis, P. (1998). A variational representation for certain functionals of Brownian motion. *The Annals of Probability*, 26(4):1641–1659.

[9] Brooks, S., Gelman, A., Jones, G., and Meng, X.-L. (2011). *Handbook of markov chain monte carlo*. CRC press.

[10] Chen, T., Liu, G.-H., and Theodorou, E. (2022). Likelihood training of schrödinger bridge using forward-backward SDEs theory. In *International Conference on Learning Representations*.

[11] Chen, T., Liu, G.-H., and Theodorou, E. A. (2021). Likelihood training of Schrödinger bridge using forward-backward SDEs theory. *arXiv preprint arXiv:2110.11291*.

[12] Cuturi, M. (2013). Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*.

[13] Dai Pra, P. (1991). A stochastic control approach to reciprocal diffusion processes. *Applied mathematics and Optimization*, 23(1):313–329.

[14] Daxberger, E. and Hernández-Lobato, J. M. (2019). Bayesian variational autoencoders for unsupervised out-of-distribution detection. *arXiv preprint arXiv:1912.05651*.

[15] De Bortoli, V., Thornton, J., Heng, J., and Doucet, A. (2021). Diffusion Schrödinger bridge with applications to score-based generative modeling. *arXiv preprint arXiv:2106.01357*.

[16] Diethe, T. (2015). 13 benchmark datasets derived from the UCI, DELVE and STATLOG repositories. https://github.com/tdiethe/gunnar_raetsch_benchmark_datasets/.

[17] Doucet, A., De Freitas, N., Gordon, N. J., et al. (2001). *Sequential Monte Carlo methods in practice*, volume 1. Springer.

[18] Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987). Hybrid Monte Carlo. *Physics letters B*, 195(2):216–222.

[19] Eches, O., Dobigeon, N., Mailhes, C., and Tourneret, J. Y. (2010). Bayesian Estimation of Linear Mixtures Using the Normal Compositional Model. Application to Hyperspectral Imagery. *IEEE Transactions on Image Processing*, 19(6):1403–1413.

[20] Edwards, H. and Storkey, A. (2016). Towards a neural statistician. *arXiv preprint arXiv:1606.02185*.

[21] Ferianc, M., Maji, P., Mattina, M., and Rodrigues, M. (2021). On the effects of quantisation on model uncertainty in Bayesian neural networks. *arXiv preprint arXiv:2102.11062*.

[22] Giles, M. (2008). An extended collection of matrix derivative results for forward and reverse mode automatic differentiation.

[23] Girolami, M. and Calderhead, B. (2011). Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(2):123–214.

[24] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

[25] Gordon, J. (2021). *Advances in Probabilistic Meta-Learning and the Neural Process Family*. PhD thesis, University of Cambridge.

[26] Gordon, J., Bronskill, J., Bauer, M., Nowozin, S., and Turner, R. E. (2018). Meta-learning probabilistic inference for prediction. *arXiv preprint arXiv:1805.09921*.

[27] Graves, A. (2011). Practical variational inference for neural networks. *Advances in neural information processing systems*, 24.

[28] Grenander, U. and Miller, M. I. (1994). Representations of knowledge in complex systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 56(4):549–581.

[29] Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR.

[30] Gyöngy, I. and Krylov, N. (1996). Existence of strong solutions for Itô's stochastic equations via approximations. *Probability theory and related fields*, 105(2):143–158.

[31] Hartmann, C., Richter, L., Schütte, C., and Zhang, W. (2017). Variational characterization of free energy: Theory and algorithms. *Entropy*, 19(11):626.

[32] Hartmann, C. and Schütte, C. (2012). Efficient rare event simulation by optimal nonequilibrium forcing. *Journal of Statistical Mechanics: Theory and Experiment*, 2012(11):P11004.

[33] Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications.

[34] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

[35] Hensman, J., Fusi, N., and Lawrence, N. D. (2013). Gaussian processes for big data. *arXiv preprint arXiv:1309.6835*.

[36] Hoffman, M. D. and Blei, D. M. (2015). Structured stochastic variational inference. In *Artificial Intelligence and Statistics*, pages 361–369.

[37] Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *Journal of Machine Learning Research*, 14(5).

[38] Hsieh, Y.-P., Kavis, A., Rolland, P., and Cevher, V. (2018). Mirrored langevin dynamics. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

[39] Huang, J., Jiao, Y., Kang, L., Liao, X., Liu, J., and Liu, Y. (2021). Schrödinger-Föllmer sampler: Sampling without ergodicity. *arXiv preprint arXiv:2106.10880*.

[40] Izmailov, P., Vikram, S., Hoffman, M. D., and Wilson, A. G. (2021). What are Bayesian neural network posteriors really like? *arXiv preprint arXiv:2104.14421*.

[41] Kappen, H. J. (2005). Linear theory for control of nonlinear stochastic systems. *Physical review letters*, 95(20):200201.

[42] Khan, M. E. and Rue, H. (2021). The bayesian learning rule. *arXiv preprint arXiv:2107.04562*.

[43] Kingma, D. P., Salimans, T., Poole, B., and Ho, J. (2021). Variational diffusion models. *arXiv preprint arXiv:2107.00630*.

[44] Kingma, D. P. and Welling, M. (2013). Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*.

[45] Klami, A., Virtanen, S., and Kaski, S. (2013). Bayesian canonical correlation analysis. *Journal of Machine Learning Research*, 14(4).

[46] Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

[47] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

[48] LeCun, Y. and Cortes, C. (2010). MNIST handwritten digit database.

[49] Léonard, C. (2012). From the Schrödinger problem to the Monge–Kantorovich problem. *Journal of Functional Analysis*, 262(4):1879–1920.

[50] Léonard, C. (2013). A survey of the Schrödinger problem and some of its connections with optimal transport. *arXiv preprint arXiv:1308.0215*.

[51] Li, X., Wong, T.-K. L., Chen, R. T. Q., and Duvenaud, D. K. (2020). Scalable gradients and variational inference for stochastic differential equations. In *Symposium on Advances in Approximate Bayesian Inference*, pages 1–28. PMLR.

[52] Maoutsa, D. and Opper, M. (2021). Deterministic particle flows for constraining SDEs. *arXiv preprint arXiv:2110.13020*.

[53] Neal, R. M. et al. (2011). Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2.

[54] Nüsken, N. and Richter, L. (2021). Solving high-dimensional Hamilton–Jacobi–Bellman PDEs using neural networks: perspectives from the theory of controlled diffusions and measures on path space. *Partial Differential Equations and Applications*, 2(4):1–48.

[55] Opper, M. (2019). Variational inference for stochastic differential equations. *Annalen der Physik*, 531(3):1800233.

[56] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

[57] Pavon, M. (1989). Stochastic control and nonequilibrium thermodynamical systems. *Applied Mathematics and Optimization*, 19(1):187–202.

[58] Pavon, M., Tabak, E. G., and Trigila, G. (2018). The data-driven Schrödinger bridge. *arXiv preprint*.

[59] Powell, W. B. (2019). From reinforcement learning to optimal control: A unified framework for sequential decisions. *CoRR*, abs/1912.03513.

[60] Pritchard, J., M., S., and P., D. (2000). Inference of population structure using multilocus genotype data. *Genetics*, 155(2):945–959.

[61] Reich, S. (2019). Data assimilation: the Schrödinger perspective. *Acta Numerica*, 28:635–711.

[62] Richter, L., Boustati, A., Nüsken, N., Ruiz, F. J., and Akyildiz, Ö. D. (2020). Vargrad: a low-variance gradient estimator for variational inference. *arXiv preprint arXiv:2010.10436*.

[63] Roberts, G. O. and Tweedie, R. L. (1996). Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, pages 341–363.

[64] Roeder, G., Wu, Y., and Duvenaud, D. (2017). Sticking the landing: Simple, lower-variance gradient estimators for variational inference. *arXiv preprint arXiv:1703.09194*.

[65] Schrödinger, E. (1932). Sur la théorie relativiste de l'électron et l'interprétation de la mécanique quantique. In *Annales de l'institut Henri Poincaré*, volume 2, pages 269–310.

[66] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[67] Thijssen, S. and Kappen, H. (2015). Path integral control and state-dependent feedback. *Physical Review E*, 91(3):032104.

[68] Tzen, B. and Raginsky, M. (2019a). Neural stochastic differential equations: Deep latent Gaussian models in the diffusion limit. *arXiv preprint arXiv:1905.09883*.

[69] Tzen, B. and Raginsky, M. (2019b). Theoretical guarantees for sampling and inference in generative models with latent diffusions. In *Conference on Learning Theory*, pages 3084–3114. PMLR.

[70] Vargas, F., Thodoroff, P., Lamacraft, A., and Lawrence, N. (2021). Solving Schrödinger bridges via maximum likelihood. *Entropy*, 23(9).

[71] Vigario, R. (1997). Meg data for studies using independent component analysis. *http://www. cis. hut. fi/projects/ica/eegmeg/MEG_data. html*.

[72] Wang, G., Jiao, Y., Xu, Q., Wang, Y., and Yang, C. (2021). Deep generative learning via Schrödinger bridge. *arXiv preprint arXiv:2106.10410*.

[73] Welling, M. and Teh, Y. W. (2011). Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688. Citeseer.

[74] Xu, W., Chen, R. T. Q., Li, X., and Duvenaud, D. (2021). Infinitely deep Bayesian neural networks with stochastic differential equations. *arXiv preprint arXiv:2102.06559*.

[75] Yoon, J., Kim, T., Dia, O., Kim, S., Bengio, Y., and Ahn, S. (2018). Bayesian model-agnostic meta-learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 7343–7353.

[76] Zhang, C., Bütepage, J., Kjellström, H., and Mandt, S. (2018). Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):2008–2026.

[77] Zhang, Q. and Chen, Y. (2021). Diffusion normalizing flow. *arXiv preprint arXiv:2110.07579*.

[78] Zhang, Q. and Chen, Y. (2022). Path integral sampler: A stochastic control approach for sampling. In *International Conference on Learning Representations*.