AdaAnn: Adaptive Annealing Scheduler for Probability Density Approximation

Emma R. Cobian, Jonathan D. Hauenstein,
Fang Liu, and Daniele E. Schiavazzi

Department of Applied and Computational Mathematics and Statistics,
University of Notre Dame, Notre Dame, IN, USA

Abstract

Approximating probability distributions can be a challenging task, particularly when they are supported over regions of high geometrical complexity or exhibit multiple modes. Annealing can be used to facilitate this task which is often combined with constant *a priori* selected increments in inverse temperature. However, using constant increments limits the computational efficiency due to the inability to adapt to situations where smooth changes in the annealed density could be handled equally well with larger increments. We introduce AdaAnn, an adaptive annealing scheduler that automatically adjusts the temperature increments based on the expected change in the Kullback–Leibler divergence between two distributions with a sufficiently close annealing temperature. AdaAnn is easy to implement and can be integrated into existing sampling approaches such as normalizing flows for variational inference and Markov chain Monte Carlo. We demonstrate the computational efficiency of the AdaAnn scheduler for variational inference with normalizing flows on a number of examples, including posterior estimation of parameters for dynamical systems and probability density approximation in multimodal and high-dimensional settings.

1 Introduction

One of the most fundamental challenges in statistics and machine learning is the ability to learn a posterior distribution from its pointwise evaluations. In this context, Markov chain Monte Carlo (MCMC) sampling is a popular paradigm to provide empirical approximations of distributions and has given rise to a large family of sampling procedures such as the Metropolis Hasting algorithm [12, 30], the Gibbs sampler [9], and slice sampling [32], among others [37, 23]. However, MCMC can be computationally expensive and may fail to capture complicated posterior distributions, leading to poor approximations.

Recently, optimization-based approaches using variational inference (VI) [5, 6, 15, 41] have emerged which aim to provide a more efficient alternative to sampling-based methods with the ability to support distributions with complex shapes such as multi-modality in high-dimensional settings [35]. More recently, VI approaches based on normalizing flows (NFs) [36], a type of generative model, are able to characterize even complex dependence in multivariate distributions. They offer a flexible framework by transforming a base distribution through a composition of invertible mappings until the desired complexity has been attained.

There are many different types of NFs such as planar flows [36], radial flows [36], real-NVP [7], autogressive flows (including inverse autoregressive flow (IAF) [19] and masked autoregressive flow (MAF) [33]), and glow [18], among others. An introduction to the fundamental principles of NFs, including their expressive power and computational trade-offs, together with a review of a wide verity of flow formulations is provided in [21, 33]. Since their introduction, they have been often combined with VI for density estimation and sampling tasks. For example, NFs are used to formulate Gaussian processes as function priors in [29], while in [24], NFs are introduced in the setting of graph neural networks for prediction and generation. In [44], NFs are applied to 3D point cloud generation; [26] applies NFs to approximate the latent variables in Bayesian neural networks. Recent applications of NFs include semi-supervised learning [14], coupling with surrogate modelling for inference with computationally expensive models [42], and solving inverse problems [43], among others.

In this study we focus on VI via NFs, specifically on situations where the target distribution to be approximated is supported over a geometrically complex subset of the parameter space or has multiple modes. Rather than designing new types of NFs offering improved representations of multimodal densities, we choose instead to approximate a collection of intermediate smoother posteriors generated through a parameterization defined in terms of an annealing temperature.

Annealing or tempering of probability density functions is used in optimization (e.g., simulated annealing [20] and simulated tempering [28]) and MCMC sampling to generate realizations from complex and multimodal distributions (e.g. tempered transition [31] and parallel tempering [10]). Tempering is also used in Bayesian statistics to study theoretical properties and concentration rates for posterior distributions [4]. This has been extended in [2] to analyze the concentration of VI approximations of (tempered) posteriors and in [13] to develop an annealed version of the objective functions in VI to improve inferential explorability. An annealed version of the free energy formulation for VI via NFs by approximating a series of tempered distributions with slowly increased inverse temperatures to provide better results on the final approximated target distribution is given in [36].

Various temperature cooling schedules have been proposed to improve computational efficiency in simulated annealing such as simple linear schedules [20], exponential multiplicative cooling [20], and logarithmic multiplicative cooling [1], among others. There also exists work on adaptive cooling where the temperature at each state transition depends on an adaptive factor based on the difference between the current solution and the best achieved solution of an objective function, including some recent work [16, 27]. Outside the realm of simulated annealing, annealing strategies and cooling schedules have received little attention.

We use a simple instance of NFs, namely, planar flows [36], to motivate our methodological development for an annealing scheduler in the settings of VI via NFs. Planar flows are shown to be a universal approximator in L_1 for one-dimensional problems in theory [22], but sometimes have been associated with a limited approximation power and more complex flow formulations have often been preferred in applications. Therefore, this is limiting the analysis of this flow in the literature, particularly for higher-dimensional latent spaces and complicated posterior distributions. We outline cases where planar flows alone fail to capture the structure of a multimodal density but the combination with annealing leads to successful approximations.

Our main contribution is *AdaAnn* (Adaptive Annealing), a novel scheduler that adaptively selects the change in temperature during the annealing process by tracking the Kullback–Leibler (KL) divergence between successive temperature changes. Through six examples of various types including multimodal distributions in high-dimensional settings, we demonstrate that AdaAnn helps NFs converge to the target posterior and leads to significant computational savings compared to a linear scheduler for all cases. In addition, we show how planar flows with AdaAnn achieve better approximation to the target distribution compared to more expressive flows without using annealing.

The remainder of the paper is organized as follows. Section 2 provides necessary background information regarding NFs and VI. Section 3 describes AdaAnn, our new adaptive annealing schedule for VI via NFs. Six examples are presented in Section 4 which demonstrate the superior performance of using annealing for VI via NFs, and the computational advantage of AdaAnn over linear annealing schedulers. We conclude with a discussion in Section 5.

2 Background

2.1 Normalizing Flows

Normalizing flows are compositions of invertible and differentiable mappings used to transform samples from a base probability density function (pdf) q_0, e.g., a standard Gaussian, into samples from a desired distribution and vice-versa. Consider a single layer of a normalizing flow with a bijection $f : \mathbb{R}^d \rightarrow \mathbb{R}^d$ that maps a set of N sample points $\{\boldsymbol{z}_0^{(i)}\}_{i=1}^N$ where $\boldsymbol{z}_0^{(i)} \sim \boldsymbol{Z}_0$, $i = 1, \dots, N$, from the base density to $\{\boldsymbol{z}_1^{(i)}\}_{i=1}^N$ where$\boldsymbol{z}_1^{(i)} = f(\boldsymbol{z}_0^{(i)})$, $i = 1, \dots, N$, and d is the dimension of $\boldsymbol{Z}_0$ and $\boldsymbol{Z}_1$. Given $\boldsymbol{Z}_0 \sim q_0$, the density of the transformed variables $\boldsymbol{Z}_1 \sim q_1$ can be computed using the change of variables formula and the properties of inverse functions, namely

$$q_1(\boldsymbol{Z}_1) = q_0(f^{-1}(\boldsymbol{Z}_1)) \cdot \left| \det \left(\frac{\partial f^{-1}}{\partial \boldsymbol{Z}_1} \right) \right| = q_0(\boldsymbol{Z}_0) \cdot \left| \det \left(\frac{\partial f}{\partial \boldsymbol{Z}_0} \right) \right|^{-1}. \tag{1}$$

One can easily generalize this to L layers of transformations so that the initial set of sample points are transformed to

$$\boldsymbol{z}_L^{(i)} = f_L \circ f_{L-1} \circ \cdots \circ f_2 \circ f_1(\boldsymbol{z}_0^{(i)}), \;\; i = 1, \dots, N, \tag{2}$$

and the corresponding pdf is given by

$$q_L(\boldsymbol{Z}_L) = q_0(\boldsymbol{Z}_0) \cdot \prod_{\ell=1}^{L} \left| \det \left(\frac{\partial f_\ell}{\partial \boldsymbol{Z}_{\ell-1}} \right) \right|^{-1}. \tag{3}$$

To simplify the computation, a desirable property of flow f_ℓ is that the Jacobian determinant is easy to compute, e.g., through the product of the diagonal entries, as in lower triangular Jacobian matrices. Many different formulations of NFs have been investigated in the literature. In this paper, we use planar flows and the real-valued Non Volume Preserving (realNVP) flows to demonstrate our proposed methodology, which are summarized next.

Planar flows [36] are one of the simpler instances of NFs where each layer transforms a set of samples with expansions or contractions perpendicular to a d-dimensional hyperplane. A planar flow $f : \mathbb{R}^d \times \mathbb{R}^{2d+1} \to \mathbb{R}^d$ consists of an activation function $h : \mathbb{R} \to \mathbb{R}$ and parameters $\phi = \{\boldsymbol{u} \in \mathbb{R}^d, \boldsymbol{w} \in \mathbb{R}^d, b \in \mathbb{R}\}$ such that:

$$f(\boldsymbol{Z}; \phi) = \boldsymbol{Z} + \boldsymbol{u} \cdot h(\boldsymbol{w}^T \boldsymbol{Z} + b). \tag{4}$$

When $\boldsymbol{u}^T \boldsymbol{w} \geq -1$, this flow is invertibile [36] and its Jacobian determinant is equal to

$$\left| \det \left(\frac{\partial f}{\partial \boldsymbol{Z}} \right) \right| = |\det\big(\boldsymbol{I} + \boldsymbol{u}\,(\boldsymbol{w} h'(\boldsymbol{w}^T \boldsymbol{Z} + b))^T\big)| = |1 + \boldsymbol{u}^T \boldsymbol{w} h'(\boldsymbol{w}^T \boldsymbol{Z} + b)|, \tag{5}$$

where h' is the derivative of h. With L layers, the transformed random variable

$$\boldsymbol{Z}_L = f_L(\bullet; \phi_L) \circ f_{L-1}(\bullet; \phi_{L-1}) \circ \cdots \circ f_2(\bullet; \phi_2) \circ f_1(\boldsymbol{Z}_0; \phi_1) \tag{6}$$

has corresponding pdf

$$q_L(\boldsymbol{Z}_L) = q_0(\boldsymbol{Z}_0) \prod_{\ell=1}^{L} |1 + \boldsymbol{u}_\ell^T \boldsymbol{w}_\ell \cdot h'(\boldsymbol{w}_\ell^T \boldsymbol{Z}_{\ell-1} + b_\ell)|^{-1}. \tag{7}$$

To enhance the expressiveness of NFs while maintaining a linear complexity in the computation of the Jacobian determinant, dependencies between different components of latent vectors $\boldsymbol{Z}_\ell$, $\ell = 1, \ldots, L$, can be introduced through autoregressive transformations. A widely used auto-regressive flow is realNVP, defined as

$$Z_{\ell+1,j} = \begin{cases} Z_{\ell,j}, & \text{for } j = 1, \ldots, c_\ell, \\ Z_{\ell,j} \exp(a_{s_k}(Z_{\ell,1}, \ldots, Z_{\ell,c_\ell})) + a_{t_k}(Z_{\ell,1}, \ldots, Z_{\ell,c_\ell}), & \text{for } j = c_\ell + 1, \ldots, d,\ k = j - c_\ell, \end{cases} \tag{8}$$

where $Z_{\ell+1,j}$ denotes the j^{th} component of $\boldsymbol{Z}_{\ell+1}$ in layer $\ell + 1$, and a_{s_k} and a_{t_k} are scale and translation functions in layer k, respectively, and are usually implemented as neural networks. The components in $\boldsymbol{Z}$ are divided into two groups in Eq. (8). The variables in the first group are copied directly into the next layer whereas the remaining variables go through an autoregressive transformation. The roles of the two groups are reversed (or the variables are randomly scrambled) after every layer. Since the c_ℓ-th component of $\boldsymbol{Z}_{\ell+1}$ in layer $\ell + 1$ depends only on the components $1, \ldots, c_\ell$ of $\boldsymbol{Z}_\ell$, the Jacobian matrix is lower triangular and its determinant is simply the product of the diagonal entries $\prod_{k=1}^{d-c_\ell} a_k(\boldsymbol{Z}_{k-1})$. In particular, realNVP is efficient and has the same computational complexity for sampling and density estimation [7]. Even if the mappings $\boldsymbol{a}_s$ and $\boldsymbol{a}_t$ are not invertible, the transformation in Eq. (8) is still invertible since

$$Z_{\ell,j} = \\ \begin{cases} Z_{\ell+1,i} & \text{for } j = 1, \ldots, c_\ell, \\ [Z_{\ell+1,j} - a_{t_k}(Z_{\ell,1}, \ldots, Z_{\ell,c_\ell})] \exp(-a_{s_k}(Z_{\ell,1}, \ldots, Z_{\ell,c_\ell})) & \text{for } j = c_\ell + 1, \ldots, d,\ k = j - c_\ell. \end{cases} \tag{9}$$

2.2 Variational Inference via Normalizing Flows

Variational inference is a common method for statistical inference and machine learning that approximates probability densities by minimizing their KL divergence from a target distribution. In particular, VI provides an effective alternative to sampling-based approaches for density approximation such as MCMC. It is based on optimization and designed to offer improved computational efficiency. Additionally, one of the major applications of NFs is VI. Without loss of generality, we illustrate the application of NFs for VI in approximating the posterior distribution $p(\boldsymbol{Z}|\boldsymbol{X})$ of the model parameters $\boldsymbol{Z}$ given observed data $\boldsymbol{X}$. Such an approximation is obtained by minimizing the free energy $\mathcal{F}$, the negative of which is a lower bound to the marginal log-density function $\log p(\boldsymbol{X})$ (a.k.a., the evidence). Due to the analytical difficulty in maximizing the marginal log-density function, the minimization of the free energy is often used in VI. If $q_\phi(\boldsymbol{Z}|\boldsymbol{X})$ is the variational distribution with parameters ϕ that approximates the true posterior $p(\boldsymbol{Z}|\boldsymbol{X})$, the free energy is

$$\begin{aligned} \mathcal{F}(\boldsymbol{X}, \phi) &= \mathbb{D}[q_\phi(\boldsymbol{Z}|\boldsymbol{X}) \,\|\, p(\boldsymbol{Z})] - \mathbb{E}_{q_{\boldsymbol{\phi}}}[\log\, p(\boldsymbol{X}|\boldsymbol{Z})] \\ &= \mathbb{E}_{q_{\boldsymbol{\phi}}}[\log q_{\boldsymbol{\phi}}(\boldsymbol{Z}|\boldsymbol{X}) - \log p(\boldsymbol{Z}, \boldsymbol{X})] \end{aligned} \tag{10}$$

where $\mathbb{D}[\cdot\|\cdot]$ denotes the KL divergence between two distributions. Following the notation in Section 2.1, we express the density $q_\phi(\boldsymbol{Z}|\boldsymbol{X})$ as $q_L(\boldsymbol{Z}_L)$ and apply the change of variables formula in Eq. (3) to Eq. (10) to obtain

$$\begin{aligned} \mathcal{F}(\boldsymbol{X}, \phi) &= \mathbb{E}_{q_0}[\log q_L(\boldsymbol{Z}_L) - \log p(\boldsymbol{X}, \boldsymbol{Z}_L)] \\ &= \mathbb{E}_{q_0}[\log q_0(\boldsymbol{Z}_0)] - \mathbb{E}_{q_0}\left[\sum_{\ell=1}^{L} \log \left| \det \frac{\partial f_\ell}{\partial \boldsymbol{Z}_{\ell-1}} \right|\right] - \mathbb{E}_{q_0}[\log(p(\boldsymbol{X}, \boldsymbol{Z}_L))]. \end{aligned} \tag{11}$$

Minimization of the free energy $\mathcal{F}$ with respect to the parameters ϕ is often achieved through gradient-based optimization, e.g., stochastic gradient descent, RMSprop [39], Adam [17], and others. The expectations in Eq. (11) are often replaced by their Monte Carlo (MC) estimates by using N realizations from the base distribution q_0. Applying Eq. (7) for planar flows, Eq. (11) becomes

$$\mathcal{F}(\boldsymbol{X}, \phi) \approx \frac{1}{N}\sum_{i=1}^{N}\Big[\log(q_0(\boldsymbol{z}_{0,i})) - \log(p(\boldsymbol{z}_{L,i}, \boldsymbol{X})) - \sum_{\ell=1}^{L}\log\big|1 + \boldsymbol{u}_\ell^T \boldsymbol{w}_\ell\, h(\boldsymbol{w}_\ell^T \boldsymbol{z}_{\ell-1,i} + b_\ell)\big|\Big]. \tag{12}$$

2.3 Annealing

Annealing (also called tempering) is a useful technique when sampling from complicated distributions by smoothing them. Coupled with MCMC techniques or VI, annealing can help improve sampling efficiency and accuracy. During the application of annealing, the inverse temperature t continuously increases with

$$p_t(\boldsymbol{Z}, \boldsymbol{X}) = p^t(\boldsymbol{Z}, \boldsymbol{X}), \text{ for } t \in (0, 1]. \tag{13}$$

The result of exponentiation by $t \in (0, 1)$ smooths out the distribution and reduces to a unimodal distribution as $t \to 0^+$. Since the original distribution is obtained as $t \to 1^-$, annealing provides a continuous deformation from an easier to approximate unimodal distribution to the original distribution.

In practice, a discrete version of Eq. (13) is used by generating a sequence of functions

$$p_k(\boldsymbol{Z}, \boldsymbol{X}) = p^{t_k}(\boldsymbol{Z}, \boldsymbol{X}), \text{ for } k = 0, \dots, K \tag{14}$$

where $0 < t_0 < \cdots < t_K \leq 1$ is an annealing scheduler and $p_k(\boldsymbol{Z}, \boldsymbol{X})$ is the annealed or tempered distribution. A commonly used annealing schedule is linear [36] of the form $t_j = t_0 + j(1 - t_0)/K$ for $j = 0, \dots, K$ with constant increments $\epsilon = (1 - t_0)/K$. For example, when combining annealing with VI and planar flows, the free energy $\mathcal{F}$ in Eq. (12) is

$$\mathcal{F}(\boldsymbol{x}, \phi) \approx \frac{1}{N} \sum_{i=1}^{N} \Big[\log(q_0(\boldsymbol{z}_{0,i})) - t_k \log(p(\boldsymbol{z}_{L,i}, \boldsymbol{X})) - \sum_{\ell=1}^{L} \log \left| 1 + \boldsymbol{u}_\ell^T \boldsymbol{w}_\ell h(\boldsymbol{w}_\ell^T \boldsymbol{z}_{\ell-1,i} + b_\ell) \right| \Big]. \tag{15}$$

2.4 A Motivating Example

Consider sampling from the pdf $p : \mathbb{R} \to \mathbb{R}$ where

$$p(Z) = 0.954 \cdot e^{-[(Z+2)^2 - 3]^2}. \tag{16}$$

Hence, $p(Z)$ is a bimodal distribution with peaks at $Z = -2 \pm \sqrt{3}$ which is the "target" in Figure 1. Consider its variational approximation $q_L(Z) \approx p(Z)$ obtained by transforming a base distribution $\mathcal{N}(\mu, \sigma^2)$ with $\mu = 0$ and $\sigma^2 = 4$ using a composition of $L = 100$ planar flow layers with hyperbolic tangent activation. We use the Adam optimizer with a learning rate of 0.005 and train 8,000 iterations consisting of $N = 100$ sample points each. In our experiment, the outcome yields Figure 1 which suggests that the optimal q_L without annealing is only able to capture a single mode. The plots at various iterations throughout this optimization can be found in Figure 22 in Section A.1 of the appendix. Using the same planar flow but with annealing as given in Eq. (15), our experiment showed that both modes were captured as shown in Figure 2(d) with a final loss[1] of 0.0024.

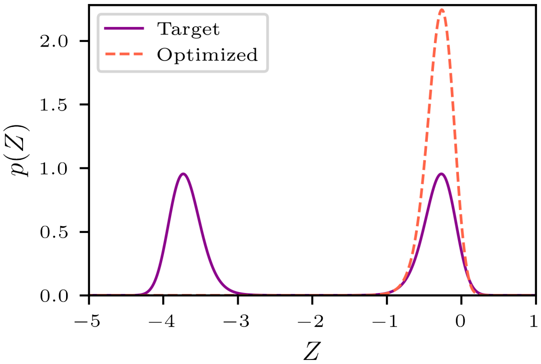
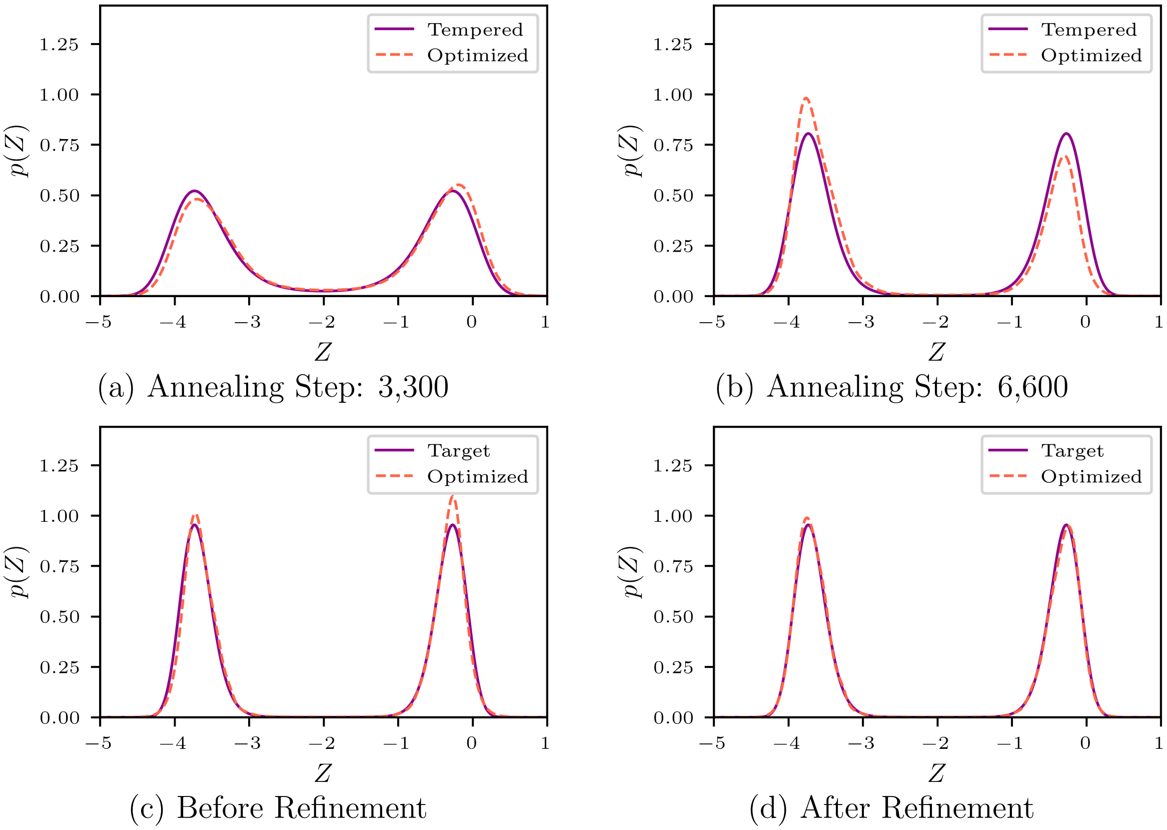


Figure 1: Variational approximation for bimodal density $p(Z)$ without annealing.

The annealing strategy used in Figure 2 had a schedule with an initial inverse temperature of $t_0 = 0.01$ that increases with a constant step size of $\epsilon = 10^{-4}$. The Adam optimizer

[1] All final loss values computed as the average of 100 MC loss values using 1,000 sample points.

ran for 500 iterations at t_0[2] and one iteration afterwards throughout the annealing process. Additional iterations were run at $t = 1$ with the number of samples points increased to $N = 1{,}000$, indicated as the *refinement training* phase.[3] The number of iterations in this phase was determined by reaching a maximum of 8,000 iterations overall or reaching the following convergence criteria: the percent change of the most recent average loss value and the previous average loss value computed every 200 iterations[4] is less than 0.5%. All together, a total of 10,400 iterations were run through NFs before the annealing temperature reached 1. Figures 2(a)-(c) show the the intermediate density approximation around one-third and two-thirds into the annealing phase and before the refinement phase when the temperature reached 1. Additionally, during refinement a step learning rate scheduler was applied which decreased the learning rate by a factor $\gamma = 0.5$ after 1,000 iterations.



(a) Annealing Step: 3,300

(b) Annealing Step: 6,600

(c) Before Refinement

(d) After Refinement

Figure 2: Variational approximation for bimodal density $p(Z)$ with linear annealing.

This relatively large number of iterations is rather typical with linear annealing schedulers to reach a variational approximation of a target distribution with satisfactory accuracy. The large number of iterations is due to the typical small steps of constant size characterizing

[2] Since the basic distribution may be significantly different than the first annealed distribution, we used a larger number of iterations at t_0.

[3] This phase allows for a more refined approximation to $p(Z)$ through an increased number of iterations and sample points.

[4] The loss function value in each iteration is the average of 50 MC loss evaluations using the current sample size.

linear annealing schedulers (e.g., 10^{-4} in the above example). An exceedingly large temperature step during the annealing process could lead to a sub-optimal approximation that does not capture the main structural features of the target distribution (e.g., missing a mode in a multi-modal distribution).

In the following, we propose a new annealing strategy that can significantly cut down the number of iterations in NFs for VI without sacrificing the quality of the final approximation.

3 Method

The following proposes the AdaAnn scheduler, a new *adaptive annealing scheduler*, that uses an adjustable step size $\epsilon_k = \epsilon_k(t) > 0, k = 1, \ldots, K$, designed to reduce the number of steps K as much as possible while providing accurate distributional approximations in VI via NFs.

3.1 AdaAnn Scheduler

Intuitively, small temperature changes are desirable to carefully explore the parameter spaces at the beginning of the annealing process, whereas larger changes can be taken as t_k increases after annealing has helped the approximate distribution to capture important features of the target distribution (e.g., locating all the relevant modes). In VI, the KL divergence based loss function in Eq. (10) can be used as a metric to adjust the annealing temperature increment. In this context, the proposed AdaAnn scheduler determines the increment ϵ_k that approximately produces a pre-defined change in the KL divergence between two distributions tempered at t_k and $t_{k+1} = t_k + \epsilon_k$, respectively. In particular, the KL divergence between these two distributions is given by

$$\mathbb{D}[p^{t_k}(\boldsymbol{Z})||p^{t_k+\epsilon_k}(\boldsymbol{Z})] = \int c(t_k)\, p^{t_k}(\boldsymbol{Z}) \log\left(\frac{c(t_k)\, p^{t_k}(\boldsymbol{Z})}{c(t_k+\epsilon_k)\, p^{t_k+\epsilon_k}(\boldsymbol{Z})}\right) d\boldsymbol{Z}, \tag{17}$$

where $c(s) = 1/\int p^s(\boldsymbol{Z})\, d\boldsymbol{Z}$ denotes the normalizing constant associated with $p^s(\boldsymbol{Z})$. A Taylor series expansion of the right hand side of Eq. (17) leads to the following theorem.

Theorem 1. *For two tempered pdfs p^{t_k} and $p^{t_k+\epsilon_k}$ with annealing step ϵ_k, the KL divergence is*

$$\mathbb{D}[p^{t_k}(\boldsymbol{Z})\|p^{t_k+\epsilon_k}(\boldsymbol{Z})] = \frac{\epsilon_k^2}{2}\, \mathbb{V}_{p^{t_k}}[\log p(\boldsymbol{Z})] + O(\epsilon_k^3) \approx \frac{\epsilon_k^2}{2} \mathbb{V}_{p^{t_k}}[\log p(\boldsymbol{Z})]. \tag{18}$$

Letting the KL divergence equal a constant $\tau^2/2$, where τ is referred to as the KL divergence tolerance, the step size ϵ_k becomes

$$\epsilon_k = \frac{\tau}{\sqrt{\mathbb{V}_{p^{t_k}}[\log p(\boldsymbol{Z})]}}. \tag{19}$$

Proof. For simplifying the presentation, we avoid using subscripts. From the definition of

KL divergence, we have

$$\begin{aligned}\mathbb{D}[p^t(\boldsymbol{Z})||p^{t+\epsilon}(\boldsymbol{Z})] &= \int c(t)\, p^t(\boldsymbol{Z})\, \log\left(\frac{c(t)\, p^t(\boldsymbol{Z})}{c(t+\epsilon)\, p^{t+\epsilon}(\boldsymbol{Z})}\right) d\boldsymbol{Z} \\ &= \int c(t)\, p^t(\boldsymbol{Z})\, \log\left(\frac{c(t)}{c(t+\epsilon)} p^{-\epsilon}(\boldsymbol{Z})\right) d\boldsymbol{Z}.\end{aligned}$$

The Taylor expansion of $c(t)/c(t+\epsilon)$ has the form

$$\begin{aligned}\frac{c(t)}{c(t+\epsilon)} &= c(t) \int p^{t+\epsilon}(\boldsymbol{Z})\, d\boldsymbol{Z} = c(t) \int p^t(\boldsymbol{Z}) \left[1 + \epsilon \log p(\boldsymbol{Z}) + \frac{[\epsilon \log p(\boldsymbol{Z})]^2}{2} + \dots\right] d\boldsymbol{Z} \\ &= c(t) \int p^t(\boldsymbol{Z})\, d\boldsymbol{Z} + c(t) \int p^t(\boldsymbol{Z})\, \epsilon\, \log p(\boldsymbol{Z})\, d\boldsymbol{Z} + c(t) \int p^t(\boldsymbol{Z}) \frac{[\epsilon \log p(\boldsymbol{Z})]^2}{2}\, d\boldsymbol{Z} + \cdots \\ &= 1 + \epsilon\, \mathbb{E}_{p^t}[\log p(\boldsymbol{Z})] + \frac{\epsilon^2}{2} \mathbb{E}_{p^t}[\log(p(\boldsymbol{Z}))^2] + O(\epsilon^3)\end{aligned}$$

and its logarithm is

$$\begin{aligned}\log\left(\frac{c(t)}{c(t+\epsilon)}\right) &= \log\Big(1 + \epsilon\, \mathbb{E}_{p^t}[\log p(\boldsymbol{Z})] + \frac{\epsilon^2}{2} \mathbb{E}_{p^t}[\log\big(p(\boldsymbol{Z})^2\big)] + O(\epsilon^3)\Big) \\ &= \epsilon\, \mathbb{E}_{p^t}[\log p(\boldsymbol{Z})] + \frac{\epsilon^2}{2}\, \mathbb{E}_{p^t}[(\log p(\boldsymbol{Z}))^2] - \frac{\epsilon^2}{2}\, \mathbb{E}_{p^t}[\log p(\boldsymbol{Z})]^2 + O(\epsilon^3) \\ &= \epsilon\, \mathbb{E}_{p^t}[\log p(\boldsymbol{Z})] + \frac{\epsilon^2}{2}\, \mathbb{V}_{p^t}[\log p(\boldsymbol{Z})] + O(\epsilon^3).\end{aligned}$$

Putting everything together with $\log p^{-\epsilon}(\boldsymbol{Z}) = -\epsilon \log p(\boldsymbol{Z})$, we have

$$\begin{aligned}\mathbb{D}[p^t(\boldsymbol{Z})||p^{t+\epsilon}(\boldsymbol{Z})] &= \int c(t)\, p^t(\boldsymbol{Z}) \left\{\epsilon\, \mathbb{E}_{p^t}[\log p(\boldsymbol{Z})] + \frac{\epsilon^2}{2} \mathbb{V}_{p^t}[\log p(\boldsymbol{Z})] + O(\epsilon^3) - \epsilon \log p(\boldsymbol{Z})\right\} d\boldsymbol{Z} \\ &= \epsilon\, \mathbb{E}_{p^t}[\log p(\boldsymbol{Z})] + \frac{\epsilon^2}{2}\, \mathbb{V}_{p^t}[\log p(\boldsymbol{Z})] - \epsilon\, \mathbb{E}_{p^t}[\log p(\boldsymbol{Z})] \\ &= \frac{\epsilon^2}{2}\, \mathbb{V}_{p^t}[\log p(\boldsymbol{Z})] + O(\epsilon^3).\end{aligned}$$

□

The quantity $\mathbb{V}_{p^{t_k}}[\log p(\boldsymbol{Z})]$ in Theorem 1 can be approximated using a MC estimate with samples from $q_L^{t_k} \approx p^{t_k}$ available from NFs at a given temperature t_k. Specifically, we draw M samples, $\boldsymbol{z}_L^{(i)}$, $i = 1, \dots, M$, and compute the sample variance of $\{\log p(\boldsymbol{z}^{(i)})\}_{i=1}^M$ using

$$S^2 = \frac{1}{M-1} \sum_{i=1}^{M} \left(\log p(\boldsymbol{z}^{(i)}) - \overline{\log p(\boldsymbol{z})}\right)^2, \text{ where } \overline{\log p(\boldsymbol{z})} = \frac{1}{M} \sum_{i=1}^{M} \log p(\boldsymbol{z}^{(i)}). \tag{20}$$

This MC approximation also provides the following intuitive interpretation of the AdaAnn scheduler from Theorem 1.

At the beginning of the annealing process, t_0 is small and the tempered distribution p^{t_0} is rather flat; the likelihood of samples coming from every region of the support of p is about the same, leading to a large variance of $\log(p)$. The combination of a large variance of $\log(p)$ with the constant τ (see Eq. (19)) results in a small annealing increment ϵ_k. As t increases, p^t becomes closer and closer to the target p and most of the samples from q_L^t fall in high-density regions of the target p. This causes the variance of $\log(p)$ to shrink, resulting in larger increments ϵ_k.

In summary, the mathematical formulation in Eq. (19) reflects the sensitivity of the annealing process in capturing the shape of the target distribution. In particular, t should increase slowly at the beginning of the annealing process due to rapid changes in the KL divergence at high temperatures, whereas the tempered distribution becomes less sensitive to temperature changes as it becomes increasingly similar to the target distribution.

Algorithm 1 summarizes the implementation of the AdaAnn scheduler with NFs with an implementation available at `https://github.com/ercobian/AdaAnn-VI-NF`.

Algorithm 1 AdaAnn Scheduler

```
**input**: initial inverse temperature $t_0^{-1}$, target distribution $p$, number of iterations $T_0$ at $t_0$, number of iterations $T_1$ at $t = 1$, number of iterations $T$ for $t \in (t_0, 1)$, number of NF samples $N$ for $t \in [t_0, 1)$, number of NF samples $N_1$ for $t = 1$, number of MC samples $M$ for calculation of $\epsilon$, KL divergence tolerance $\tau$, a prespecified NF structure with $L$ layers of transformation.
**output**: approximated distribution $q_L$ for $p$.
$t \leftarrow t_0$; $\epsilon \leftarrow 0$
**while** $t + \epsilon < 1$ **do**
    $t \leftarrow t + \epsilon$
    Obtain an empirical approximation $q^t$ to $p^t$ with $N$ samples using NF for the specified
        number of iterations at $t$ ($T_0$ for $t = t_0$ and $T$ for $t \in (t_0, 1)$)
    Calculate the MC estimate of $\mathbb{V}_{p^t}[\log p(\boldsymbol{Z})]$ in Eq. (19) using $\boldsymbol{z}^{(i)} \sim q^t, i = 1, \ldots, M$
        samples in Eq. (20)
    $\epsilon \leftarrow \tau / S$
**end while**
$t \leftarrow 1$
(Optional) Refine at $t = 1$ by running the NFs for $T_1$ iterations to obtain a final approximation $q$ to $p$ with $N_1$ samples.
```

4 Numerical Examples

We apply AdaAnn to six examples, where the target distributions range from bimodal univariate distribution to multimodal high-dimensional distributions, and compare its approximation accuracy and computational efficiency with linear annealing. Specifically, we first compare AdaAnn to linear schedulers in one-dimensional settings with bimodal distributions. We then examine two-dimensional bimodal densities and compare the performance of a planar flow with AdaAnn and a flow with greater approximation power (i.e., realNVP). Next, we consider two applications in dynamical systems, where we obtain posterior variational inference of the parameters of a Lorenz attractor and a non-linear dynamical system simulating HIV viral dynamics, respectively, via NFs with AdaAnn. In example 6, we compare AdaAnn and linear schedulers in a high-dimensional setting with for both unimodal and bimodal posteriors. For all examples, unless otherwise noted, we use hyperbolic tangent activation functions in planar flows, optimize the free energy loss function in VI via NFs using Adam, use the convergence criteria as in Section 2.4, and run computations on an AMD EPYC 7532 32-Core Processor.

4.1 Example 1: One-dimensional Bimodal Distribution

We apply AdaAnn to the bimodal density in Eq. (16) and compare with the linear annealing scheduler in Section 2.4. The same planar flow as specified in Section 2.4 is employed. We use Algorithm 1 with the following hyperparameters: $t_0 = 0.01$ (identical to the linear scheduler), $T_0 = 500$, $T = 2$, $T_1 = 8{,}000$, $\tau = 0.005$ and $M = 1{,}000$. For the Adam optimizer, we apply the same learning rate schedule as in Section 2.4. The number of points in each iteration increases from $N = 100$ to $N_1 = 1{,}000$ during the refinement stage at $t = 1$, and we take T_1 as the maximum number of iterations unless the convergence criteria has been met. We present in Figure 3 the distribution approximately one-third and two-thirds in the annealing phase, at $t = 1$ before refinement, and the final optimized variational distribution via the planar flow with AdaAnn. The results suggest an accurate approximation of the target distribution with a final loss value of 0.0019. The AdaAnn scheduler took 636 steps, with 1,768 parameter updates before refinement, whereas the linear scheduler took 9,902 steps as presented in Figure 4. The rate of change in t_k is slow for AdaAnn when t_k is small and increases as t_k becomes larger; this adaptive behavior helps drive the computational cost down for AdaAnn.

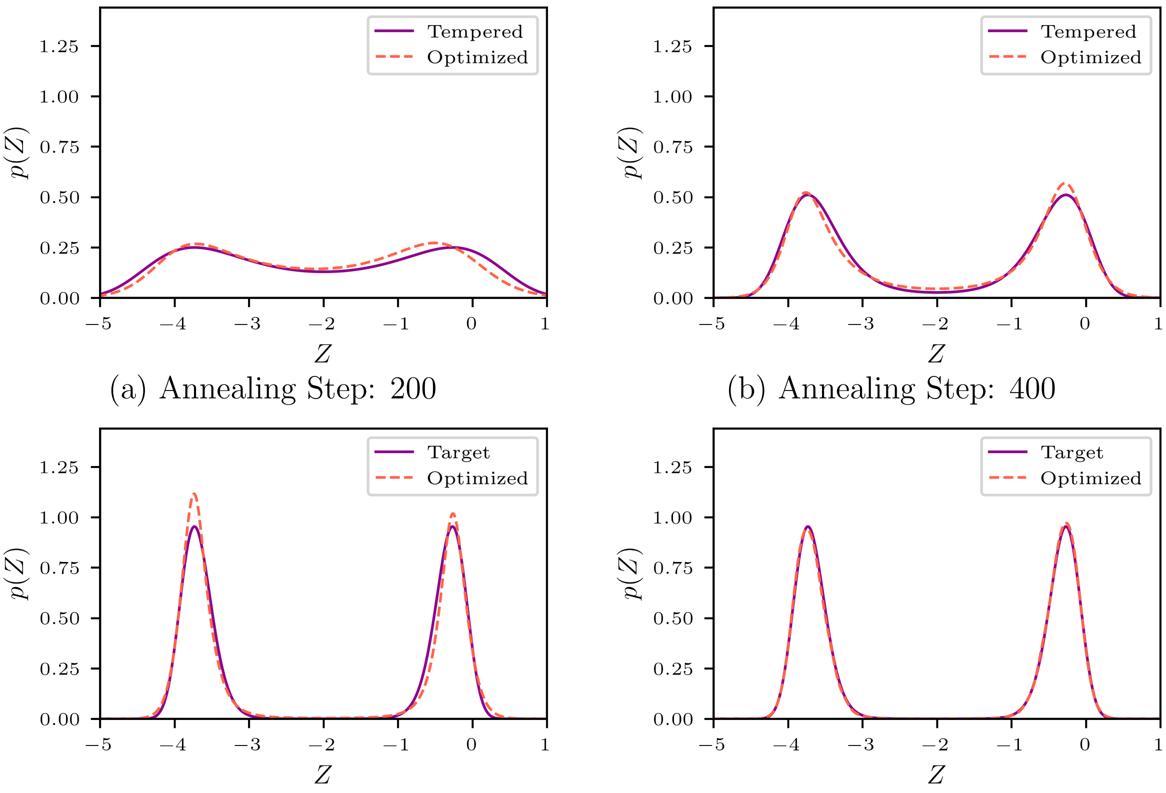
We repeated this optimization 50 times and summarized the results in Table 1. Though both AdaAnn and the linear scheduler perform well in approximating the target distribution, the computational cost associated with the linear scheduler is much higher. The total parameter updates and computational time (found in Table 2) for this example is approximately half when using AdaAnn, compared to the linear schedule, while maintaining comparable accuracy as indicated by the final loss values. These computations were performed on an Intel(R) Xeon(R) CPU E5-2680 v3 @ 2.50GHz Haswell processors with 256 GB of RAM.

Table 1: Metrics comparing AdaAnn and linear schedules over 50 trials in Example 1.

	AdaAnn		**Linear**	
Metric	Mean	SD	Mean	SD
Loss	0.0038	0.0022	0.0032	0.0033
Annealing Steps	636	14	9,902	-[5]
Refinement Iterations	6,708	1,971	6,800	2,256
Total Parameter Updates	8,476	1,972	17,200	2,256

We also examine how the choice of τ in AdaAnn affects the approximation quality and computational complexity. Toward that end, we set the KL divergence tolerance τ at four

[5] Since all trials in the linear scheduler use 9,902 annealing steps, SD is not meaningful.



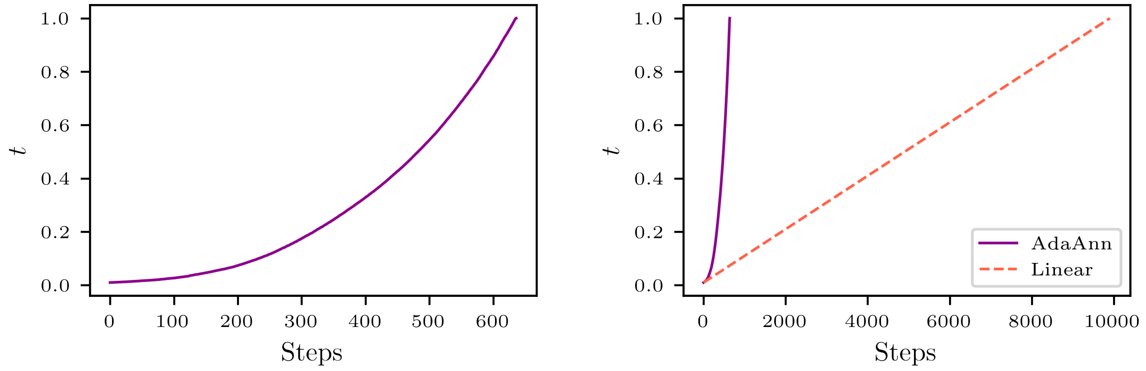
(a) Annealing Step: 200

(b) Annealing Step: 400

(c) Before Refinement

(d) After Refinement

Figure 3: Variational approximation of $p(Z)$ in Example 1 with AdaAnn scheduler.



(a) AdaAnn

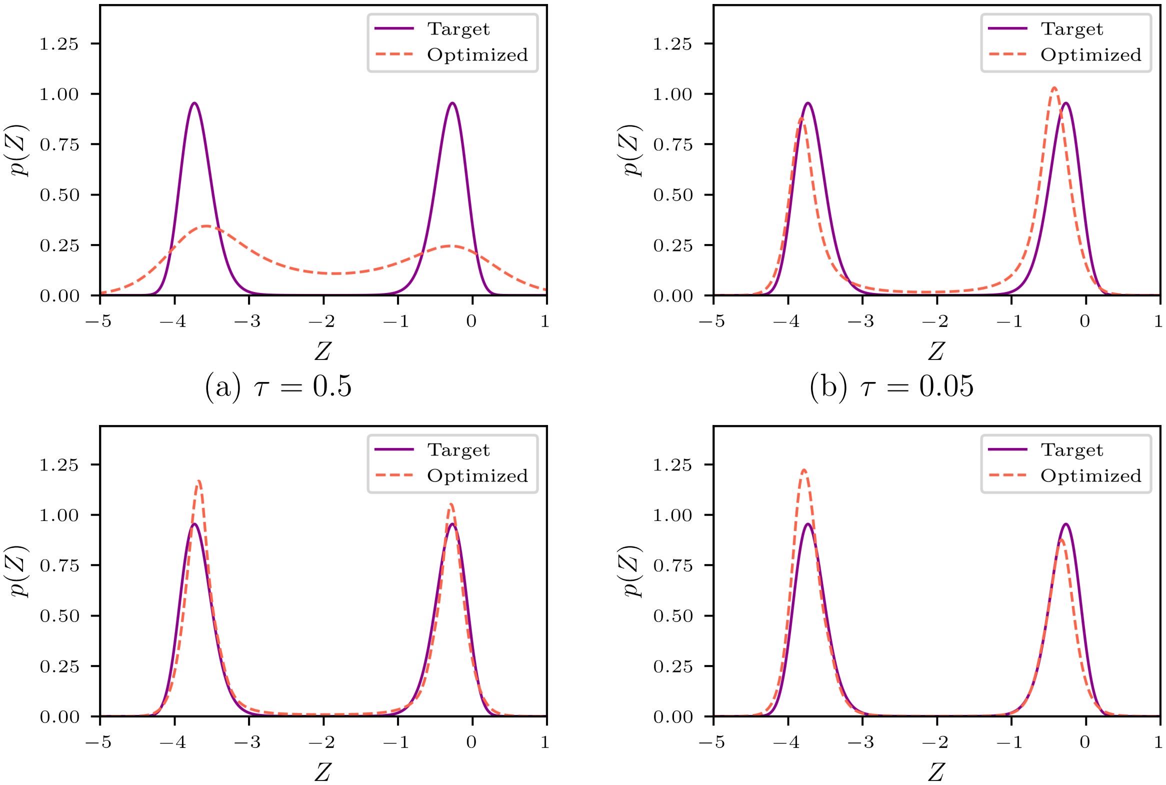
(b) AdaAnn and Linear

Figure 4: (a) AdaAnn annealing schedule and (b) comparison between the AdaAnn and a linear schedule for density approximation in Example 1.

different values (0.5, 0.05, 0.005, 0.0005) and obtained the final approximate distribution to the target in each case before refinement. The results are presented in Figure 5. Although all tolerance choices maintain the bimodal structure after further refinement, certain values provide a better approximation to the target density at the conclusion of the annealing phase. For $\tau = 0.5$, the annealing phase completed in only 14 incremental temperature steps but needs significant refinement to capture the height of the peaks. While $\tau = 0.05$ captured the

Table 2: Computational time (in minutes) for VI via NFs in Example 1, Example 2 (the symmetric case at $m = 4$), and Example 3 ($m = 4$).

Example	**NF procedure**	**5th-Percentile**	**Median**	**95th-Percentile**
1	AdaAnn	5.14	11.46	12.15
	Linear	12.82	20.35	22.14
2	No Annealing ($L = 25$)	2.80	2.84	2.97
	No Annealing ($L = 50$)	5.63	5.71	6.24
	No Annealing ($L = 100$)	10.87	11.28	11.98
	AdaAnn	4.00	6.82	8.59
	Linear	5.97	8.17	10.72
3	AdaAnn	5.53	8.36	12.12
	Linear	7.66	10.37	14.61



(a) $\tau = 0.5$

(b) $\tau = 0.05$

(c) $\tau = 0.005$

(d) $\tau = 0.0005$

Figure 5: Comparison of optimized distributions from NFs with AdaAnn at $t = 1$ for various KL tolerances τ without refinement.

peaks better in 83 annealing steps, a decent amount of refinement is still needed. For the two smaller values $\tau = 0.005$ and $\tau = 0.0005$, both capture the features of the target density while taking 659 and 5,496 steps, respectively. While $\tau = 0.0005$ provides a slightly better approximation than $\tau = 0.005$, it takes significantly more steps (8.34 folds more) without significantly improving the quality of the resulting approximation.

In summary, this example illustrates that while AdaAnn and the linear annealing schedule lead to favorable approximations to the target distribution, AdaAnn significantly reduces the number of steps needed to the final approximation and ultimately reduces the computational time. In addition, the choice of the KL divergence tolerance τ is critical for the accuracy of the variational approximation: too large of a τ value can be too crude to capture important characteristics of a distribution and too small a τ value may incur additional computational costs without significantly improving the approximation.

4.2 Example 2: One-dimensional Mixture Gaussian Distribution

We consider a mixture of two Gaussian distributions in this example, namely

$$p(Z) = \frac{1}{2\sqrt{\pi/8}} e^{-8(Z+m_1)^2} + \frac{1}{2\sqrt{\pi/8}} e^{-8(Z+m_2)^2}. \tag{21}$$

Here, $p(Z)$ depends upon two parameters, m_1 and m_2, which are varied to investigate how the distance between the two modes of $p(Z)$ and their location relative to the mode of the base distribution $q_0 = \mathcal{N}(\mu, \sigma^2)$ with $\mu = 0$ and $\sigma^2 = 16$ impact the accuracy of the optimal variational approximation. We examine two cases of $p(Z)$: (1) when the two modes are symmetrically located around 0, the mode of q_0; that is, $m_1 = -m_2$, and (2) when one of the modes is fixed at 0. We refer to these two cases as *symmetric* and *asymmetric*, respectively, and use a single parameter m to denote the distance between the two modes in both cases. For the symmetric case, we set $m_1 = m/2$ and $m_2 = -m/2$ while, for the asymmetric case, $m_1 = m$ and $m_2 = 0$. An example for each of the two cases is provided in Figure 6.

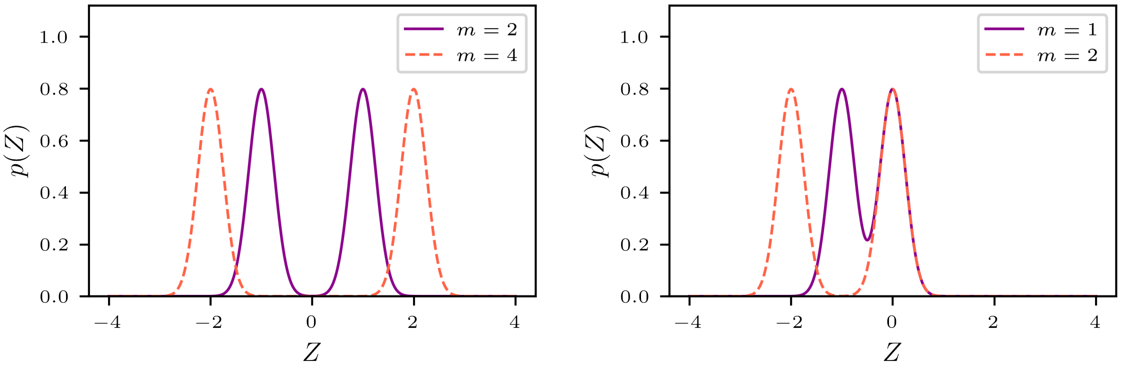


Figure 6: Illustration of symmetric and asymmetric bimodal distributions in Example 2.

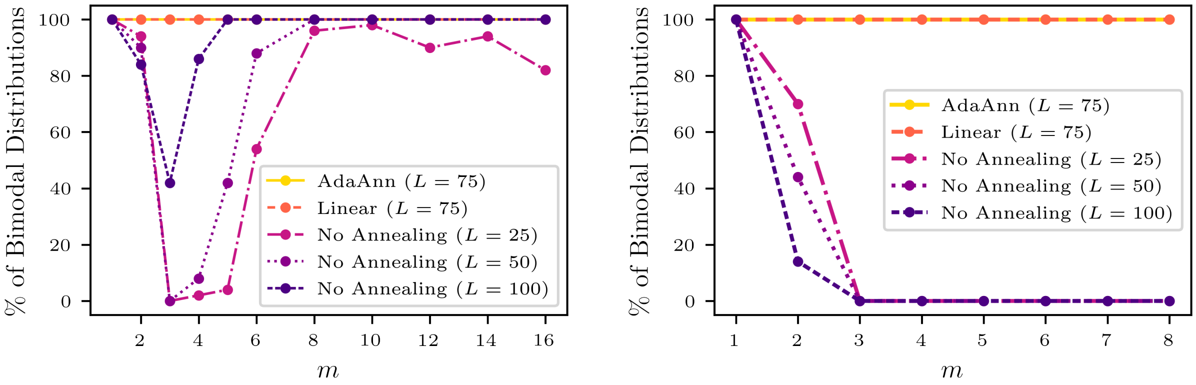
We vary m from 1 to 16 in the symmetric case and 1 to 8 in the asymmetric case. For each value of m, we run 50 trials without annealing, with a linear annealing schedule, and with the AdaAnn scheduler to approximate the target distribution for VI via NFs. The number of layers for the planar flow with annealing is $L = 75$. For the scenario without annealing, we also vary the number of layers and consider $L = 25$, 50, 100 and train the planar flow for 8,000 iterations with $N = 100$ samples per iteration. We set $t_0 = 0.01$, $T_0 = 500$, $T = 4$, $T_1 = 8{,}000$, $\tau = 0.002$, $M = 1{,}000$, $N = 100$, and $N_1 = 1{,}000$ for AdaAnn (Algorithm 1).

For the linear scheduler, we set $\epsilon = 10^{-4}$ and $T = 1$ while keeping t_0, T_0, T_1, N, and N_1 defined the same as in the AdaAnn scheduler. We again cease refinement when either the convergence criteria has been met or T_1 iterations has been reached (only in the annealing scenarios). Also, a step learning rate scheduler is applied during refinement decreasing the learning rate by a factor of $\gamma = 0.8$ every 500 iterations. The learning rates for the Adam optimizer are reported in Table 3.

Table 3: Learning rates for different values of m used in Example 2.

Symmetric Case ($m_1 = m/2, m_2 = -m/2$)		**Asymmetric Case** ($m_1 = m, m_2 = 0$)	
m	Learning Rate	m	Learning Rate
1, 2	0.02	1, 2	0.01
3, 4, 5	0.001	3	0.002
6, 8, 10, 12, 14, 16	0.0005	4, 5, 6, 7, 8	0.001

We examine how likely an approximated distribution using VI via NFs captures the bimodal structure in $p(Z)$ with the results summarized in Figure 7. We also compare the computational time required by NFs without annealing using varying number of layers (computations were performed on an Intel(R) Xeon(R) CPU E5-2680 v3 @ 2.50GHz Haswell processors with 256 GB of RAM) and the proposed AdaAnn scheduler versus a linear scheduler.



(a) Symmetric

(b) Asymmetric

Figure 7: Percentage of approximate distributions from VI via NFs which capture the bimodal target structure out of 50 trials in Example 2.

First, for the symmetric case with $m = 4$, an increasing number of planar flow layers is associated with a higher likelihood to capture the target bimodal structure without annealing (Figure 7), but at a higher computational cost (Table 2). Second, without annealing, the symmetric modes are well recovered for $m > 8$; for the asymmetric case, the two modes are captured 100% of the time only when $m = 1$, independent of the number of layers, and the percentage decreases with m until it reaches zero for $m \geq 3$. Third, with annealing, the target distributions can be accurately approximated for all the examined values of m in both the symmetric and asymmetric cases. Considering the symmetric case without annealing, there is a large drop in the percentage of recovered bimodal distributions near $m = 4$ for $L = 25$ and 50. When the modes of the target distribution are *connected*, i.e., not separated by a segment of zero probability, NFs easily captures both modes. When the modes become separated, NFs no longer capture both modes consistently. This is indicative of a rough loss landscape where the optimizer is unable to determine the global minimum. As these modes become further separated, NFs improve in capturing both modes, likely indicating the loss landscape has become smoother and the global minimum is easier to attain. Although both the linear and AdaAnn annealing schedules are able to produce bimodal approximations of similar accuracy, AdaAnn requires significantly fewer parameter updates, as summarized in Table 4 for the symmetric case; similar for the asymmetric case.[6]

Table 4: Mean and standard deviation (SD) of final loss and total parameter updates for AdaAnn and linear schedulers in the symmetric case of Example 2.

	Final Loss				**Total Parameter Updates**			
	AdaAnn		**Linear**		**AdaAnn**		**Linear**	
m	Mean	SD	Mean	SD	Mean	SD	Mean	SD
1	0.0009	0.0009	0.0007	0.0009	11,160	2,128	17,520	2,038
2	0.0019	0.0013	0.0020	0.0021	12,011	1,600	17,000	2,518
3	0.0267	0.0185	0.0302	0.0561	11,091	2,602	14,848	2,916
4	0.0246	0.0123	0.0278	0.0317	11,380	2,649	14,900	2,804
5	0.0228	0.0147	0.0317	0.0383	11,682	2,451	14,932	2,870
6	0.0390	0.0132	0.0304	0.0149	10,436	2,635	13,904	2,571
8	0.0414	0.0185	0.0313	0.0181	10,742	2,611	14,764	2,670
10	0.0409	0.0155	0.0356	0.0187	10,553	2,520	14,128	2,579
12	0.0444	0.0206	0.0346	0.0168	10,701	2,465	14,216	2,550
14	0.0538	0.0184	0.0500	0.0758	10,296	2,421	13,940	2,491
16	0.0555	0.0177	0.0400	0.0188	10,470	2,390	14,216	2,606

In summary, the results suggest that for NFs without annealing: (1) the relative location of the base distribution with respect to the locations of the modes of the target distribution may affect the accuracy of the variational approximation and (2) when the location of the base distribution is strongly biased toward one of the modes of the target distribution, successful approximation may only occur when the modes are not separated ($m \leq 1$ in this example). Annealing helps to mitigate both problems.

4.3 Example 3: Two-dimensional Bimodal Distribution

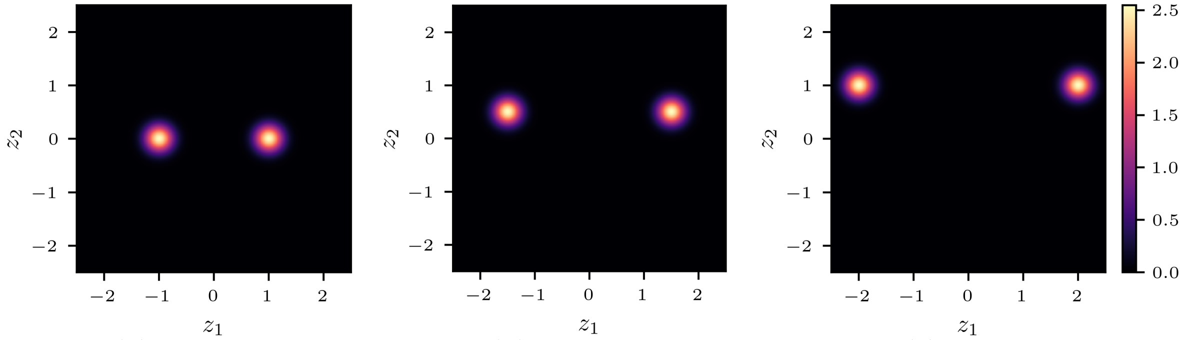
In the third example, we compare the density approximation performance between planar flows coupled with annealing and a more expressive flow such as realNVP. The target distri-

[6] Table 12 in Section A.2 of the appendix, contains these results for the asymmetric case.

bution is a mixture of two bivariate Gaussian densities expressed as

$$p(Z_1, Z_2) = \frac{8}{\pi} e^{-16\,[(Z_1+m/2)^2+(Z_2-m/2+1)^2]} + \frac{8}{\pi} e^{-16\,[(Z_1-m/2)^2+(Z_2-m/2+1)^2]}. \tag{22}$$

Here, $p(Z_1, Z_2)$ depends upon a parameter m which is used to move the modes. In particular, this density is similar to the bimodal symmetric density from Section 4.2 and has narrow modes equally spaced from the origin. As m increases, the modes will move diagonally up and away from the origin resulting in a larger separation, as seen in Figure 8.

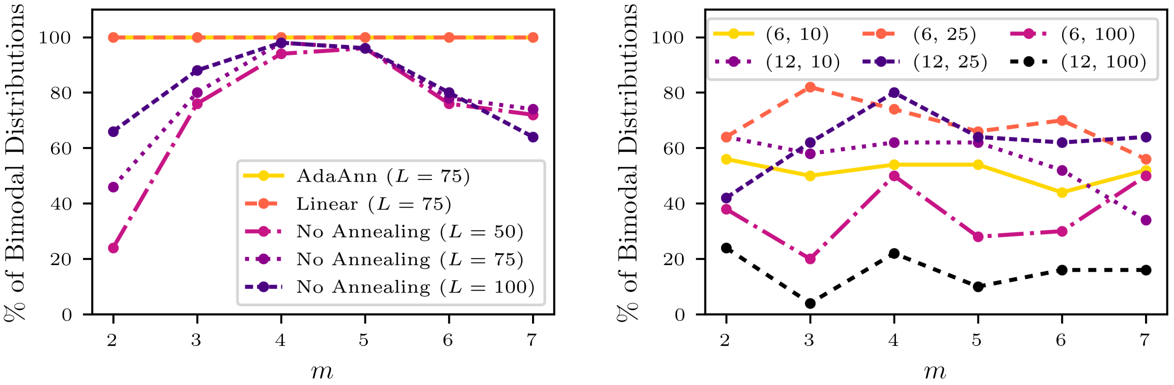


(a) $m = 2$ (b) $m = 3$ (c) $m = 4$

Figure 8: Bivariate Gaussian mixture densities for increasing values of m in Example 3.

To approximate the target distribution $p(Z_1, Z_2)$, we transform a distribution $q_0 = \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ with $\boldsymbol{\mu} = (0, 0)$ and $\Sigma = 4I_2$ using four different configurations: (1) three planar flows consisting of $L = 50, 75, 100$ layers without annealing trained for 5,000 iterations; (2) a planar flow with $L = 75$ layers combined with the AdaAnn scheduler ($t_0 = 0.01$, $T_0 = 500$, $T = 3$, $T_1 = 8{,}000$, $\tau = 0.002$, $M = 1{,}000$, $N = 100$, $N_1 = 1{,}000$); (3) a planar flow with 75 layers combined with a linear scheduler ($t_0 = 0.01$, $T_0 = 500$, $T = 1$, $T_1 = 8{,}000$, $\epsilon = 10^{-4}$, $N = 100$, $N_1 = 1{,}000$); and (4) realNVP without annealing. Both annealing schedulers apply a step learning rate scheduler with $\gamma = 0.9$ every 1,000 iterations. For realNVP, the scale and translation functions $\boldsymbol{a}_s$ and $\boldsymbol{a}_t$, respectively, consist of fully connected neural networks with two neurons for both the input and output layers, two hidden layers with H hidden neurons, and the ReLU activation function. A hyperbolic tangent activation function is applied right before the output layer on the scale function $\boldsymbol{a}_s$. We examined three cases for H, namely 10, 25, and 100. We examined two scenarios of coupling layers, namely 6 and 12, and use alternating masking that switches the variables being updated at each coupling layer. We trained the realNVP for 5,000 iterations. For the Adam optimizer, we used a batch size of $N = 100$ and the learning rates at $m = 2$ to 7 are 0.001 0.0008, 0.0005, 0.0005, 0.0005, and 0.0002 respectively.

For each m and each NFs setup, we conducted 50 trials and recorded how many times the bimodal structure of the target distribution is captured in the final optimized distribution. The results are summarized in Figure 10. In particular, both of the annealing methods capture the bimodal structure in all 50 trials at every m, outperforming the planar flows without annealing, which is consistent with the results from Examples 1 and 2. RealNVP, despite having a more complicated structure than planar flow, still fails to capture both modes in a considerable number of repetitions, suggesting that the approximation accuracy resulting from planar flow plus an annealing schedule may not be achieved by a more expressive flow alone.



(a) Planar Flow

(b) RealNVP (# of layers, # of neurons)

Figure 9: Percentage of successful distribution reconstruction from VI via NFs out of 50 trials in Example 2.

Figure 10: Percentage of successful distribution reconstruction in Example 3.

Both annealing methods with planar flows achieve the comparable accuracy, but AdaAnn has significantly fewer parameter updates during the annealing phase yielding fewer total parameter updates (with the defined stopping criteria) as presented in Table 5. This leads to superior computational efficiency, which is shown in Table 2 for $m = 4$.

Table 5: Comparing mean and standard deviation of final loss and total parameter updates for AdaAnn and linear schedulers using planar flows in Example 3.

	Final Loss				**Total Parameter Updates**			
	AdaAnn		**Linear**		**AdaAnn**		**Linear**	
$\boldsymbol{m}$	Mean	SD	Mean	SD	Mean	SD	Mean	SD
2	0.0460	0.0529	0.0446	0.0539	10,251	2,701	14,508	2,864
3	0.0335	0.0146	0.0250	0.0088	10,897	2,666	15,016	2,664
4	0.0459	0.0215	0.0416	0.0246	10,407	2,654	14,472	2,818
5	0.0461	0.0191	0.0436	0.0163	10,453	2,853	14,036	2,767
6	0.0413	0.0191	0.0439	0.0218	11,147	2,688	14,284	3,010
7	0.1044	0.0394	0.0828	0.0281	11,363	2,577	13,736	2,617

4.4 Example 4: Lorenz Attractor

After considering closed-form distributions in the first three examples, we investigate the ability of VI via NFs with annealing to solve inverse problems involving dynamical systems.

In such cases, evaluating the posterior distribution at a single realization of the input parameters (up to a constant) necessitates the numerical solution of a system of ordinary differential equations (ODEs). Specifically, in this section we consider the Lorenz attractor [25]:

$$\begin{cases} \dot{x} = s(y - x) \\ \dot{y} = x(r - z) - y \\ \dot{z} = xy - bz. \end{cases} \tag{23}$$

This system of ODEs results from a simplified representation of Rayleigh-Bénard convection and is derived from a Galerkin projection of a system of coupled Navier-Stokes and heat transfer equations with thermal convection and buoyancy. It models convection between two horizontal plates with the lower plate uniformly warmer than the upper plate. Described by this system, x is proportional to the intensity of the convective motion, y is proportional to the temperature difference between ascending and descending currents, and z is proportional to the discrepancy between the vertical temperature distribution in the model and a linear profile [25]. Restricted to positive values, s is the Prandtl number, r is the Rayleigh number, and b is a geometric factor, i.e., the aspect ratio of the convection vortices [25, 38]. The system is unstable for $\sigma > (b+1)$ and $r > r_c \approx 24.74$. In particular, for $s = 10$, $b = 8/3$, and $r = 28$, it follows a chaotic butterfly-like dynamics revolving around two strange attractors. Starting from almost identical initial conditions ($\Delta = 10^{-6}$), the system is known to generate chaotic trajectories for $t > 15$ [40].

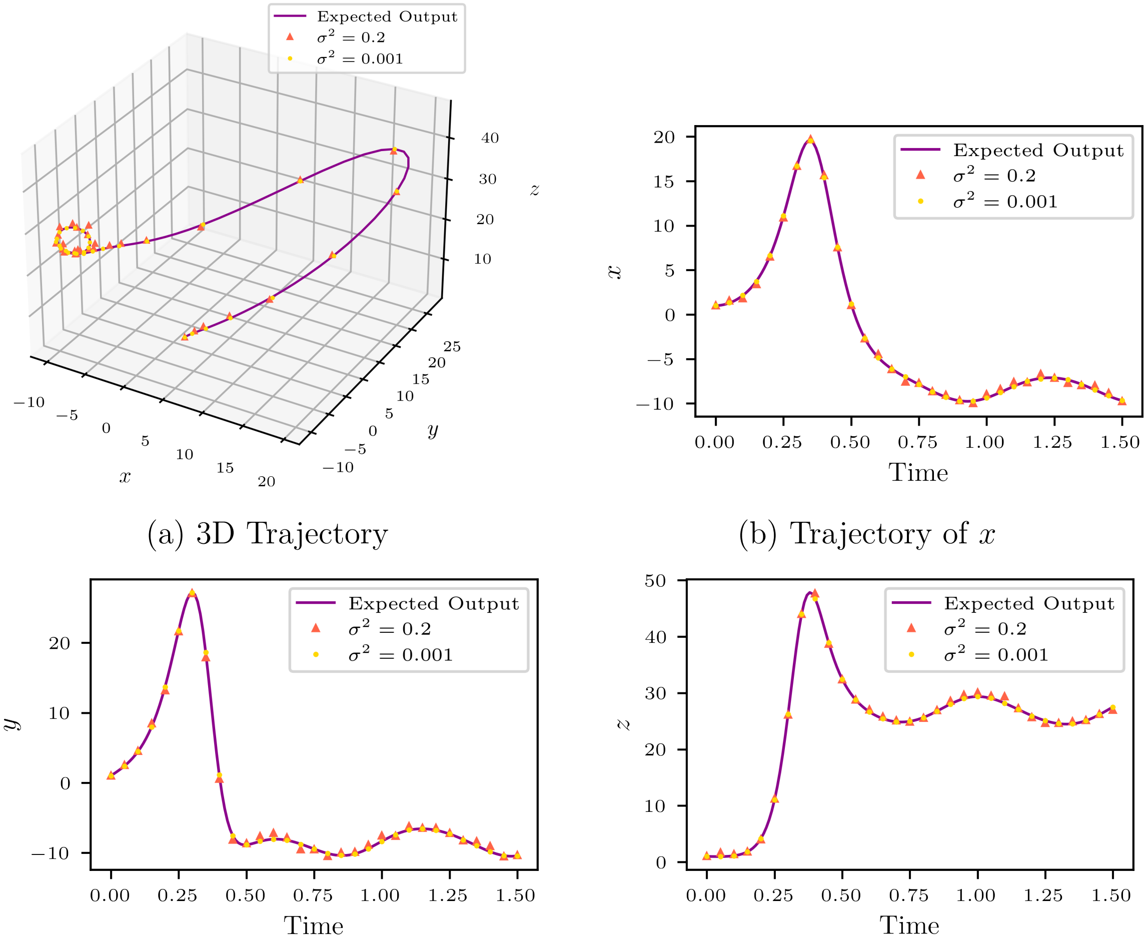
The parameters s, b, and r in Eq. (23) are often of inferential interest given a set of observations on x, y, and z. We use VI via NFs to estimate s, b, and r in a Bayesian framework. Specifically, we simulate observations given $s = 10$, $b = 8/3$, and $r = 28$ as follows. Using a fourth order Runge-Kutta method (RK4) with initial conditions $x_0 = y_0 = z_0 = 1$, the Lorenz equations are integrated in time from $t = 0$ to $t = 1.5$ with step size $\Delta t = 0.025$. From this solution $[(x_i, y_i, z_i)]_{i=1}^{60}$, we choose $n = 30$ equally spaced data points and add Gaussian noise $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$ where $\boldsymbol{\mu} = (0, 0, 0)$ and $\Sigma = \sigma^2 I_3$ with $\sigma^2 = 0.001$ and $\sigma^2 = 0.2$, generating two sets of noisy (x, y, z) realizations as shown in Figure 11.

The following is the posterior distribution of the parameters $\boldsymbol{\theta} = \{s, b, r\}$ with a non-informative uniform prior on the parameters and Gaussian likelihood function:

$$p(\boldsymbol{\theta}|(x, y, z)) \propto \frac{1}{\sqrt{(2\pi\sigma^2)^{D\cdot n}}} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} \left\| (x_i, y_i, z_i)^T - \boldsymbol{G}_i(\boldsymbol{\theta}) \right\|_2^2 \right). \tag{24}$$

The operator $\boldsymbol{G}$ outputs the RK4 solution of the Lorenz equations with respect to the input parameters $\boldsymbol{\theta}$, $D = 3$ is the dimension of the output, and $n = 30$ as above.

Starting with a base $q_0 = \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ with $\boldsymbol{\mu} = (10, 10, 10)$ and $\Sigma = 4I_3$, we use planar flow with $L = 250$ layers and apply the AdaAnn scheduler in Algorithm 1 with the following hyperparameters: $t_0 = 0.05$, $T_0 = 500$, $T = 5$, $T_1 = 5{,}000$, $\tau = 0.2$, $M = 100$, $N = 100$, and $N_1 = 200$. For the linear scheduler, we set $\epsilon = 10^{-4}$ and $T = 1$, while keeping the remaining parameters identical to the AdaAnn scheduler. The learning rate for the Adam optimizer at $t < 1$ is 0.005; during the refinement phase at $t = 1$, the batch size is increased to $N_1 = 200$ and a step learning rate scheduler is applied with a reduction of $\gamma = 0.75$ every 500 training



(a) 3D Trajectory

(b) Trajectory of x

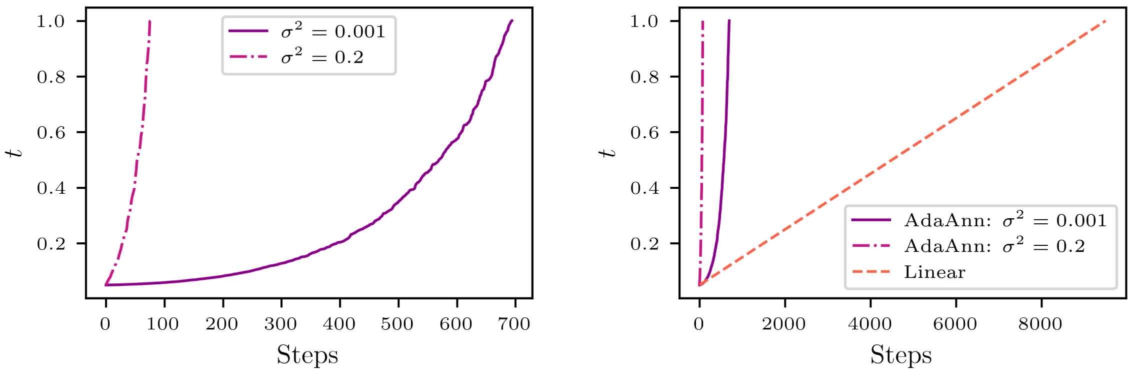
(c) Trajectory of y

(d) Trajectory of z

Figure 11: Trajectories of the Lorenz system and observations (x, y, z).

iterations. The annealing schedules are shown in Figure 12, where AdaAnn took 694 and 76 steps for $\sigma^2 = 0.001$ and $\sigma^2 = 0.2$, respectively, while the linear scheduler took 9,502 steps. Using the convergence criteria, AdaAnn had an additional 1,800 refinement iterations leading to 5,760 total parameter updates, while the linear scheduler had an additional 800 iterations and 10,800 total updates for $\sigma^2 = 0.001$. This leads to significant computational savings with the optimization completing in 80 minutes when employing the AdaAnn scheduler, versus 112 minutes for the linear scheduler.

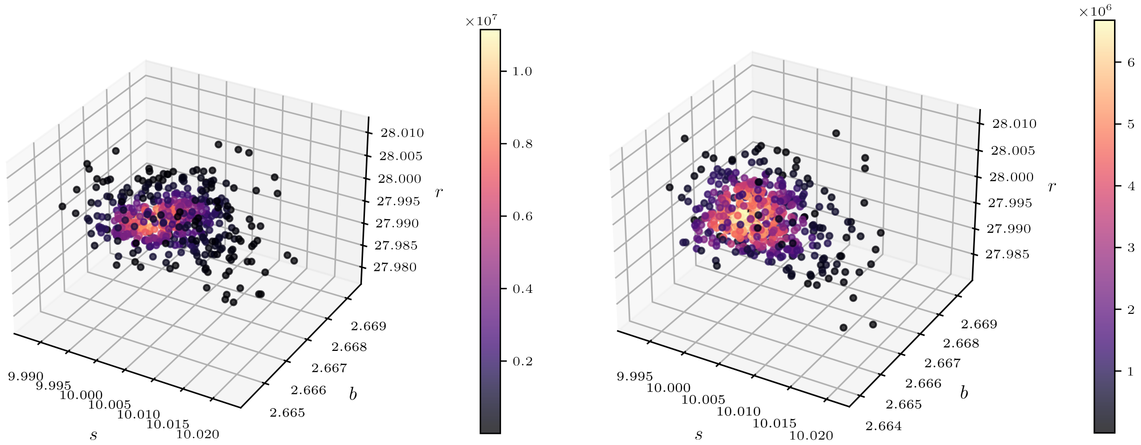
The resulting variational approximation $q_L(s, b, r|\boldsymbol{X})$ is shown in Figure 13, for $\sigma^2 = 0.001$, comparing both schedulers. The marginal histogram for each of the three parameters and the pairwise scatter plots are depicted in Figure 14. The inferred distributions agree well with the true parameter values for both AdaAnn and the linear scheduler. The MC estimates for the marginal means and standard deviations (SD) of the posterior distributions are computed from the final optimized approximate distribution q_L using 10,000 samples and reported in Table 6. We also used Monte Carlo integration to approximate the “true” mean and SD of the posterior distribution, referenced as MC True, using 10,000 points (see Table 14). For



(a) AdaAnn (b) AdaAnn and Linear

Figure 12: (a) AdaAnn annealing schedule and (b) comparison between the AdaAnn and a linear schedule for density approximation in Example 4.

each parameter, its true value is within one SD of the corresponding estimated parameter value. The corresponding plots for $\sigma^2 = 0.2$ can be found in Figures 23 and 24 in Section A.3 of the appendix, and the posterior estimates can be found in Table 13.



(a) AdaAnn (b) Linear

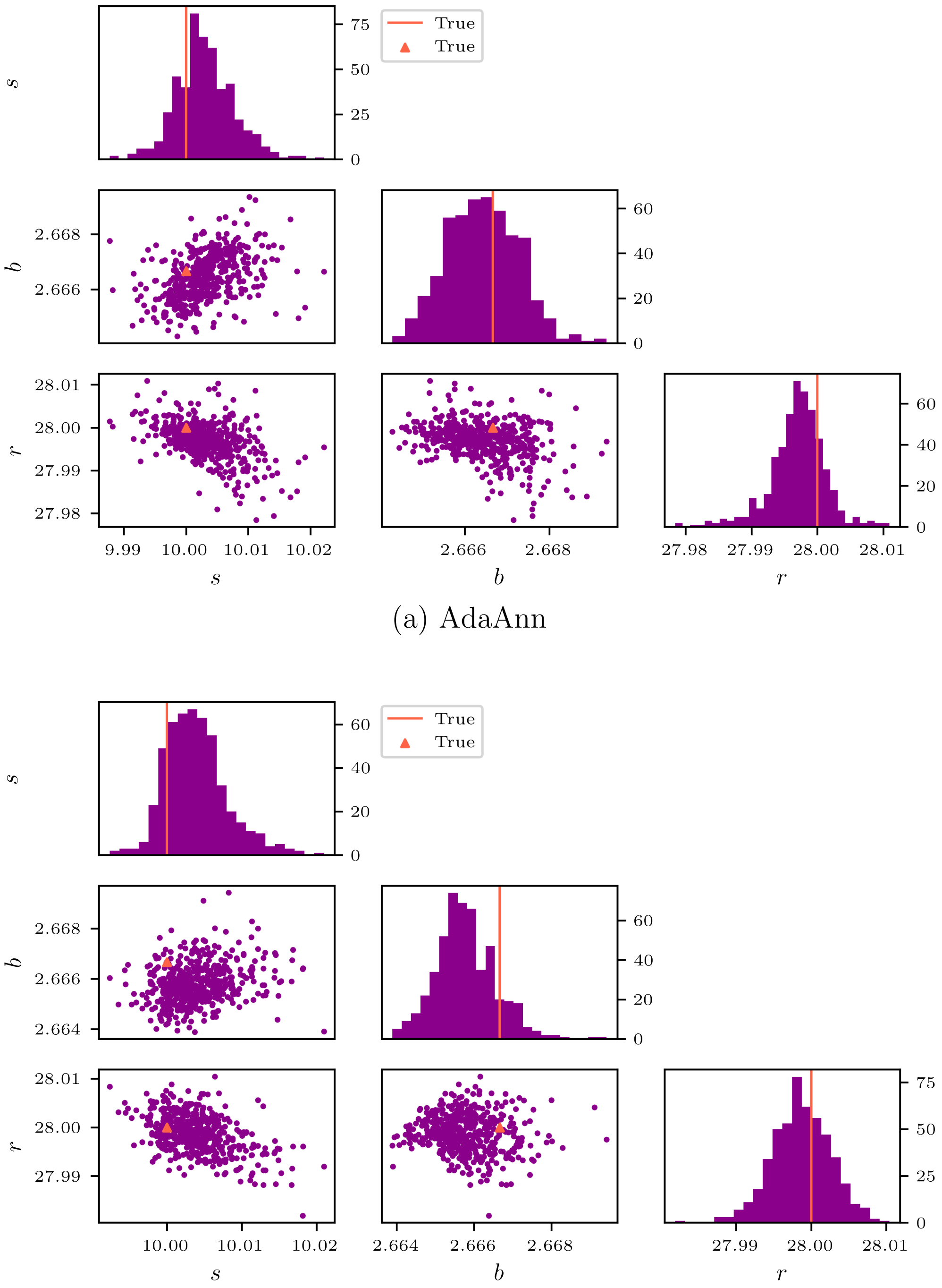
Figure 13: Approximate posterior distribution of parameters (s, b, r) for the Lorenz attractor obtained by VI via NFs with $\sigma^2 = 0.001$ using AdaAnn and linear schedulers.

4.5 Example 5: ODE system for HIV dynamics

This example infers the parameters of a system of ODEs that models the HIV dynamics [3] based on the original system [34]:

$$\begin{aligned} &\dot{x_1} = p_1 - p_2 x_1 - p_3 x_1 x_3, \quad \dot{x_2} = p_3 x_1 x_3 - p_4 x_2, \quad \dot{x_3} = p_1 p_4 x_2 - p_5 x_3, \\ &y = x_3. \end{aligned} \tag{25}$$

In this system, x_1 is the number of $CD4^+$ T-cells that are susceptible to being infected by the HIV-1 virus and x_2 is the number of productively infected $CD4^+$ T-cells. The concentration



(a) AdaAnn

(b) Linear

Figure 14: Marginal posterior distributions and pairwise scatter plots of parameters (s, b, r) for the Lorenz attractor with $\sigma^2 = 0.001$ using AdaAnn and linear schedulers.

of HIV-1 free virus, x_3, is measured in HIV-1 RNA per mL of plasma. The dynamics of the system are driven by the following five parameters: p_1 is the rate of target cells being produced from a source, p_2 is the rate of target cells dying, p_3 is the rate of target cells being

Table 6: Posterior mean and standard deviation for the parameters of the Lorenz attractor, based on 10,000 samples obtained by VI via NFs with $\sigma^2 = 0.001$ using AdaAnn and linear schedulers, and "true" posterior values using Monte Carlo integration.

True	**MC True**		**AdaAnn**		**Linear**	
Value	Post. Mean	Post. SD	Post. Mean	Post. SD	Post. Mean	Post. SD
$s = 10$	10.0028	0.0044	10.0028	0.0052	10.0038	0.0045
$b = 8/3$	2.6663	0.0008	2.6663	0.0009	2.6658	0.0009
$r = 28$	27.9986	0.0037	27.9970	0.0046	27.9989	0.0040

infected by the HIV-1 virus, p_4 is the death rate of productively infected cells x_2, and p_5 is the clearance rate of infectious HIV-1 virus particles from the body.

We use VI via NFs to estimate the parameters p_1 and p_2 along with the initial condition x_{2_0}. The remaining parameters and initial conditions are considered known and fixed. The posterior distribution of p_1 and p_2 may have a multimodal structure if this system has an identifiability degree greater than one [3]. In fact, for this problem, the identifiability degree is 2 indicating two sets of parameter values producing an identical output, namely $\{p_1, p_2, x_{2_0}\}$ and $\{-p_1, p_2, -x_{2_0}\}$, generating the same observed trajectory on output $y(t) = x_3(t)$.

The system in Eq. (25) is numerically integrated using RK4 until time $t = 2$ months with step size of $\Delta t = 0.05$ months using the parameters and initial conditions in Table 7. Synthetic data were generated using $n = 40$ equally spaced data points from $y = x_3$ and adding Gaussian errors $\mathcal{N}(\mu, \sigma^2)$ with $\mu = 0$ and $\sigma^2 = 0.0005$ to the output solution x_3, as shown in Figure 15.

Table 7: Parameter values and initial conditions in the HIV dynamics ODE system.

Unknown Parameters	**Known Parameters**	**Fixed Initial Conditions**
$p_1 = 1.2$	$p_3 = 4.1$	$x_{1_0} = 0$
$p_2 = 0.8$	$p_4 = 10.2$	$x_{3_0} = 1$
$x_{2_0} = 1.5$	$p_5 = 2.6$	

The posterior distribution of parameters $\boldsymbol{\theta} = \{p_1, p_2, x_{2_0}\}$ given n observed data points on x_3 is $p(\boldsymbol{\theta}|\boldsymbol{x}_3) \propto (2\pi\sigma^2)^{-n/2} \exp\left(\sigma^{-2}\sum_{i=1}^{n}(x_{3i} - G_i(\boldsymbol{\theta}))^2\right)$ given the Gaussian likelihood function and a uniform prior on $\boldsymbol{\theta}$. To approximate this posterior, we transform a base distribution $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$ with $\boldsymbol{\mu} = (0, 0, 0)$ and $\Sigma = 4I_3$ using a composition of $L = 250$ planar flows. We run AdaAnn with $t_0 = 0.00005$, $T_0 = 1{,}000$, $T = 5$, $\tau = 0.005$, $M = 100$, and $N = 100$. The learning rate for the Adam optimizer is 0.0005. Once we reach $t = 1$, we refine the posterior approximation by training for an additional $T_1 = 5{,}000$ iterations (without the convergence criteria), increasing the batch size to $N_1 = 200$, and adopting a step learning rate scheduler for the Adam optimizer (with the learning rate reduced by a factor of $\gamma = 0.75$ after 1,000 training iterations).

The AdaAnn schedule is depicted in Figure 16 with a total of 4,645 steps. We also overlay the

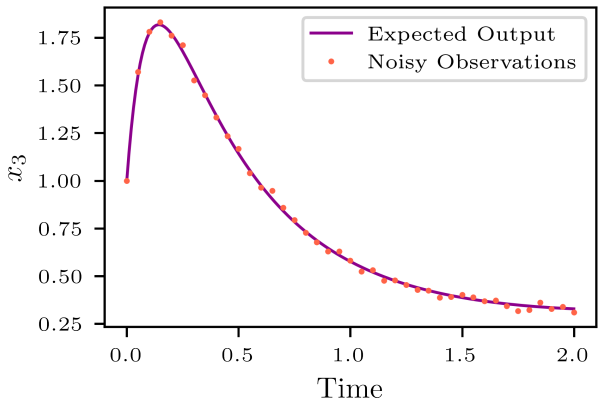


Figure 15: Trajectory of the output x_3 in units 10^4 HIV RNA per mL of plasma over 2 months along with noisy data in example 5.

adaptive step size ϵ_k using a running average over the past 40 steps. The plot of ϵ_k illustrates how AdaAnn is able to adaptively modulate the increments in inverse temperature, and therefore differs substantially from an exponential scheduler. The resulting approximation q_L captures the bimodal structure of the target posterior distribution, as presented in Figure 17. For comparison, we also run the planar flow without annealing for 20,000 iteration. The resulting q_L inconsistently converges to either a unimodal or bimodal approximation. Since only the mode with positive parameters is biologically relevant, converging to one with negative parameters may lead to the conclusion that the model is unable to reproduce the observed behavior with physically sound parameters.

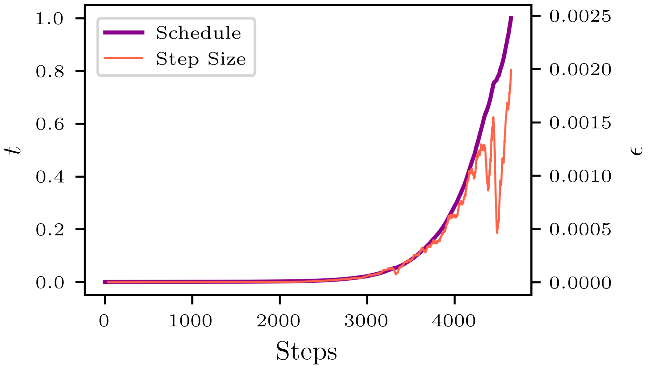
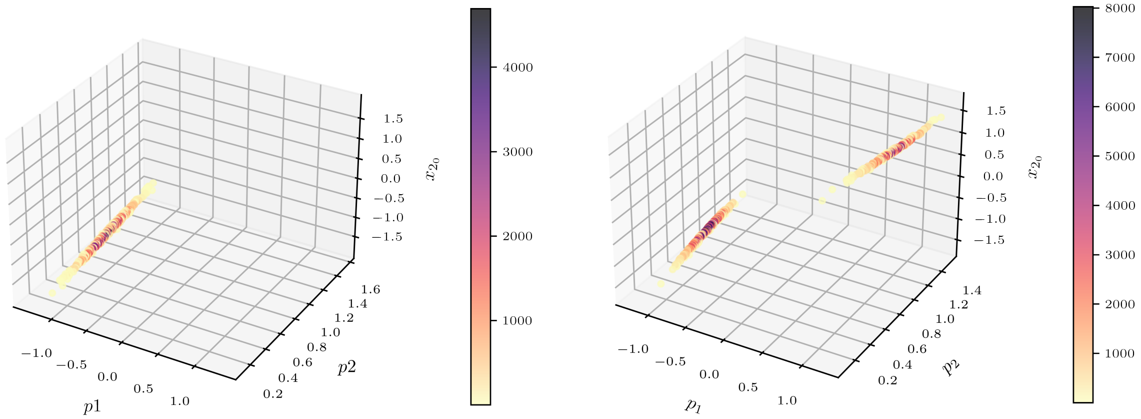


Figure 16: AdaAnn schedule and running average of adaptive step size in Example 5.

The marginal distributions are also shown in Figure 18. Since the left mode is not biologically meaningful due to negative parameter values, we also included the marginal distributions for the right mode plotted against the true parameter values in Figure 19. The true model parameters are accurately inferred by combining VI and NFs with the proposed adaptive annealing schedule. The posterior marginal means and standard deviations are computed using 10,000 samples and displayed in Table 8, along with the MC integrated “true” values.



(a) No-Annealing

(b) AdaAnn

Figure 17: Approximate posterior distributions without annealing versus with AdaAnn in Example 5.

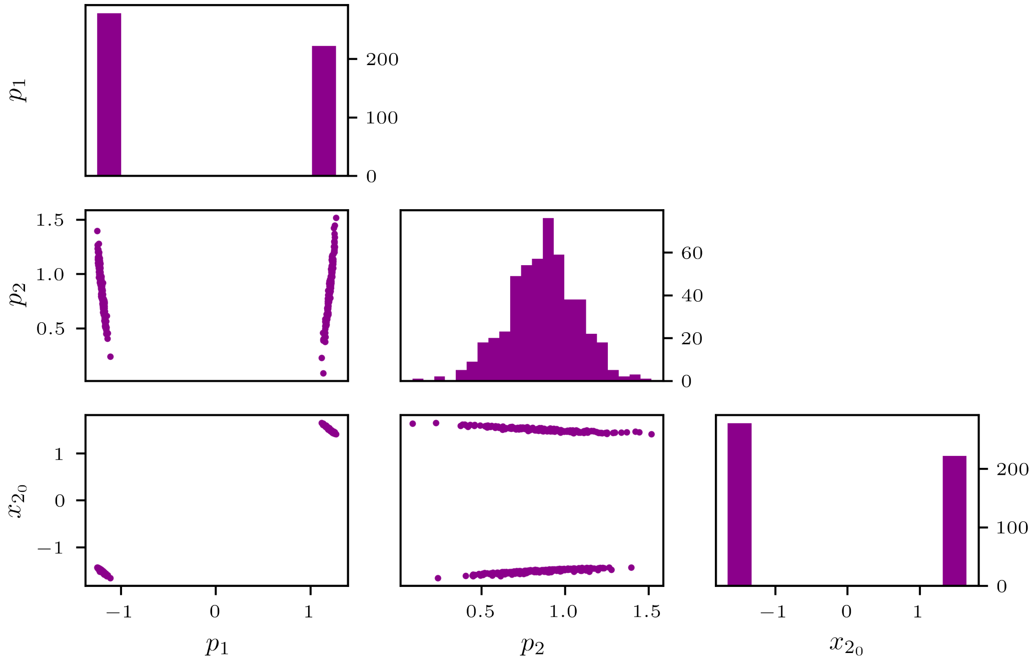


Figure 18: Marginal distributions for the HIV dynamics ODE system.

4.6 Example 6: Friedman 1 Data

We use a Friedman 1 dataset [8] to examine the performance of AdaAnn in a high-dimensional setting. The model that a Friedman 1 dataset is simulated from is given in Eq. (26)

$$y_i = \mu_i(\boldsymbol{\beta}) + \epsilon_i,\ \text{where}\ \mu_i(\boldsymbol{\beta}) = \beta_1 \text{sin}(\pi x_{i1} x_{i2}) + \beta_2 (x_{i3} - \beta_3)^2 + \textstyle\sum_{j=4}^{10} \beta_j x_{ij}, \tag{26}$$

$\boldsymbol{\beta} = (\beta_1, \ldots, \beta_{10}) = (10, 20, 0.5, 10, 5, 0, 0, 0, 0, 0)$ and $\epsilon_i \sim \mathcal{N}(0, 1)$. This model contains linear, non-linear, and interaction terms of the input variables X_1 to X_{10}, five of which (X_6

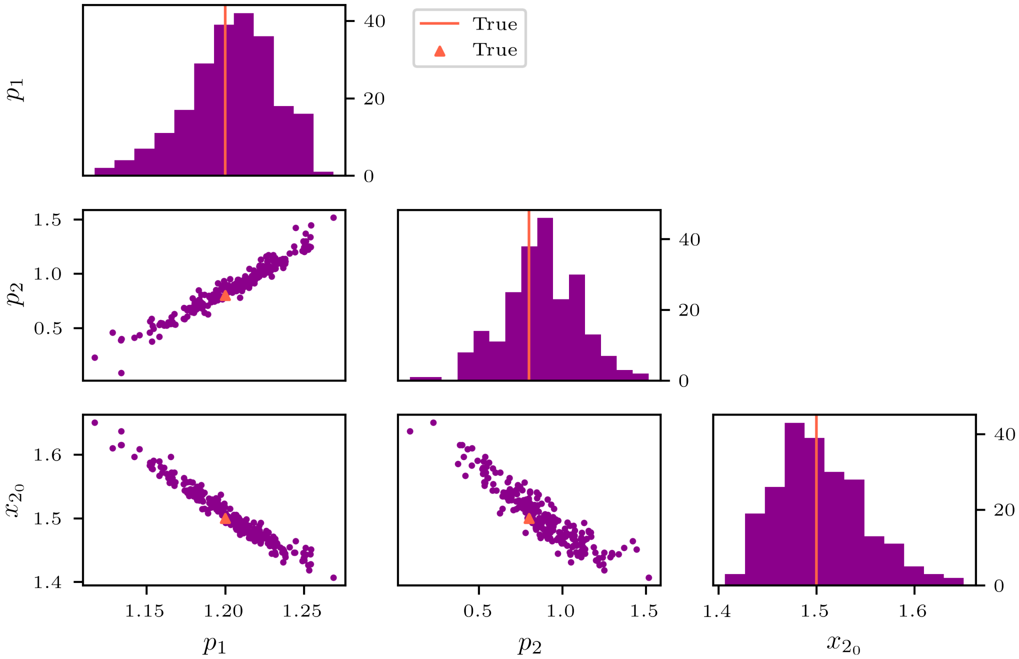


Figure 19: Marginal distribution of the biologically admissible mode for the HIV dynamics ODE system.

Table 8: Posterior mean and standard deviation of the parameters for the HIV dynamics ODE system (Example 5).

Mode	**True**	**MC True**		**AdaAnn**	
Type	**Value**	Post. Mean	Post. SD	Post. Mean	Post. SD
Biologically Admissible	$p_1 = 1.2$	1.2015	0.0169	1.2019	0.0274
	$p_2 = 0.8$	0.8520	0.1348	0.8609	0.2196
	$x_{2_0} = 1.5$	1.5048	0.0277	1.5060	0.0452
Biologically Unadmissible	$p_1 =$ -1.2	-1.2017	0.0172	-1.2014	0.0249
	$p_2 = 0.8$	0.8545	0.1385	0.8485	0.2001
	$x_{2_0} =$ -1.5	-1.5049	0.0280	-1.5067	0.0412

to X_{10}) are irrelevant to Y. Each X is drawn independently from $\mathcal{U}(0, 1)$. We used R package `tgp` [11] to generate a Friedman 1 dataset with a sample size of $n = 1{,}000$. We assume the variance of the error term ϵ, which is 1, is known, and apply VI via NFs to obtain posterior inferential on $\boldsymbol{\beta}$ given the simulated data. With the Gaussian likelihood constructed from the model in Eq. (26) and a non-informative prior $p(\boldsymbol{\beta}) \propto$ constant, the posterior distribution of $\boldsymbol{\beta}$ is

$$p(\boldsymbol{\beta}|\mathbf{y}) = \prod_{i=1}^{n} \exp\left\{-(y_i - \mu_i(\boldsymbol{\beta}))^2/2\right\}. \tag{27}$$

Starting with a base distribution $\mathcal{N}(\mathbf{0}, 4I_{10})$, we apply planar flows with $L = 250$ layers using both AdaAnn and linear schedulers. For AdaAnn, we set the following hyperparameters: $t_0 = 0.05$, $T_0 = 500$, $T = 5$, $T_1 = 5{,}000$, $\tau = 0.2$, $M = 1{,}000$, $N = 100$, and $N_1 = 1{,}000$. For the linear scheduler, we set $\epsilon = 0.001$ and $T = 1$, while the remaining hyperparameters are defined the same as in the AdaAnn scheduler. The learning rate for the Adam optimizer starts at 0.005 and decreases by a factor of $\gamma = 0.5$ every 1,000 iterations when reaching the refinement phase. The AdaAnn scheduler took 145 steps, leading to 715 parameter updates between t_0 and $t = 1$ as seen in Figure 20(a). In this case, we chose an epsilon value that lead to a similar number of parameter updates in the annealing phase for the linear scheduler. With 951 steps, this scheduler performed 6,449 total parameter updates versus 6,215 total updates when using AdaAnn (the convergence criteria was not used). In this set-up, both schedulers approximated the values for $\boldsymbol{\beta}$ similarly, as reported in Table 9.

Table 9: Posterior mean and standard deviation of the parameters for unimodal posterior in Friedman 1 data (Example 6).

True	**AdaAnn**		**Linear**	
Value	Post. Mean	Post. SD	Post. Mean	Post. SD
$\beta_1 = 10$	9.9856	0.0921	9.9863	0.0931
$\beta_2 = 20$	20.3344	0.4140	20.3426	0.3976
$\beta_3 = 0.5$	0.4982	0.0028	0.4981	0.0028
$\beta_4 = 10$	10.1329	0.1059	10.1274	0.1060
$\beta_5 = 5$	5.0234	0.1081	5.0306	0.1067
$\beta_6 = 0$	0.0621	0.1051	0.0617	0.1044
$\beta_7 = 0$	-0.0384	0.1014	-0.0378	0.1023
$\beta_8 = 0$	-0.0858	0.1060	-0.0899	0.1055
$\beta_9 = 0$	-0.0667	0.1090	-0.0674	0.1079
$\beta_{10} = 0$	0.0112	0.0998	0.0125	0.1010

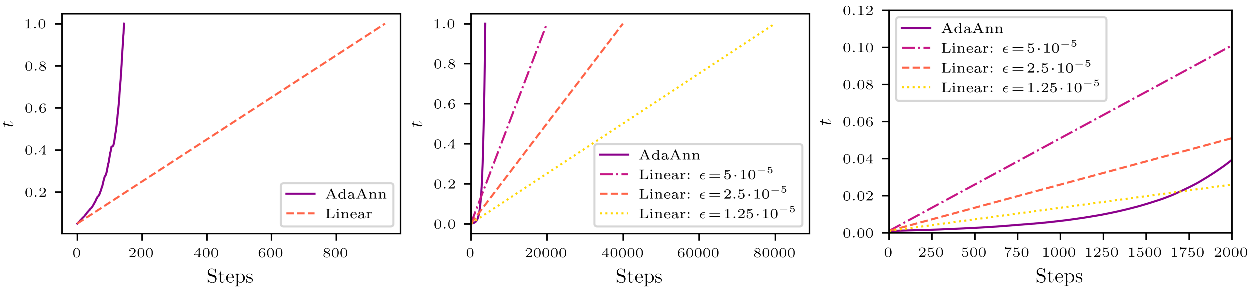
We made a slight modification to the model in Eq. (26) that generated the Friedman 1 dataset by setting

$$\mu_i(\boldsymbol{\beta}) = \beta_1 \sin(\pi x_{i1} x_{i2}) + \beta_2^2 (x_{i3} - \beta_3)^2 + \textstyle\sum_{j=4}^{10} \beta_j x_{ij}, \tag{28}$$

where $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_{10}) = (10, \pm\sqrt{20}, 0.5, 10, 5, 0, 0, 0, 0, 0)$. Again, we impose a non-informative prior $p(\boldsymbol{\beta}) \propto$ constant; though the formulation of the posterior distribution on $\boldsymbol{\beta}$ is the same as in Eq. (27) with $\mu_i(\boldsymbol{\beta})$ defined in Eq. (28), the posterior distribution of $\boldsymbol{\beta}$ is now bimodal. We approximated this bimodal distribution using NFs with a similar set-up as the unimodal Friedman 1 case with the following changes: $t_0 = 0.001$, $\tau = 0.005$, and $T_0 = 1{,}000$. When applying the AdaAnn scheduler, both modes were recovered with 3,787 annealing steps. The MC estimated values for $\boldsymbol{\beta}$ are reported in Table 10. We chose a linear scheduler with approximately the same number of parameter updates as in the annealing phase of AdaAnn (18,925 updates) and examined three settings of ϵ ($5 \times 10^{-5}, 2.5 \times 10^{-5}$, and 1.25×10^{-5}). The linear scheduler (19,980 updates) failed to recover both modes at all three ϵ values. The annealing schedules over the first 2,000 steps in Figure 20(c) suggests that the AdaAnn scheduler begins with a significantly smaller step size than any of the linear schedulers, ultimately leading to recovering both modes in this high-dimensional setting. After the bimodal structure is obtained, the AdaAnn schedule quickly increases given it adaptive nature and surpasses the linear schedules, as depicted in Figure 20(b).

Table 10: Posterior mean and standard deviation of the parameters for bimodal posterior in Example 6.

True	**Mode 1**		**Mode 2**	
Value	Post. Mean	Post. SD	Post. Mean	Post. SD
$\beta_1 = 10$	9.9865	0.0901	9.9829	0.0920
$\beta_2 = \pm\sqrt{20}$	4.5095	0.0461	-4.5070	0.0456
$\beta_3 = 0.5$	0.4978	0.0027	0.4978	0.0027
$\beta_4 = 10$	10.1330	0.1049	10.1255	0.1051
$\beta_5 = 5$	5.0273	0.1058	5.0289	0.1038
$\beta_6 = 0$	0.0594	0.1043	0.0572	0.1022
$\beta_7 = 0$	-0.0419	0.1023	-0.0299	0.1024
$\beta_8 = 0$	-0.0883	0.1052	-0.0827	0.1052
$\beta_9 = 0$	-0.0715	0.1055	-0.0665	0.1060
$\beta_{10} = 0$	0.0162	0.1008	0.0104	0.1027



(a) Unimodal (b) Bimodal (c) Bimodal zoomed in

Figure 20: Annealing schedules for the unimodal and bimodal posteriors in Friedman 1 data using AdaAnn and linear schedulers.

4.7 Summary of the Examples

The target distributions in these six examples are of varying degrees of complexity and AdaAnn produces distinct annealing schedules that are well adapted to the complexity of the underlying posterior distribution. This is evident from Figure 21 that illustrates the evolution of the inverse temperature generated by AdaAnn for Examples 1, 4, 5, and 6. A larger noise variance in the data for the Lorenz system (i.e., $\sigma^2 = 0.2$) leads to a wider posterior distribution that AdaAnn is able to approximate in few, mainly large, steps. A reduced variance ($\sigma^2 = 0.001$) corresponds instead to a more sharply peaked posterior which requires more small increments near the beginning. For the bimodal HIV dynamics posterior in 3D, characterized by two well separated peaks, AdaAnn requires significantly more steps and a smaller initial temperature, as expected. It is also interesting to observe that, in the schedule for the HIV dynamical system example, ϵ_k is reduced after $\sim$ 4,500 iteration, producing a small but visible "kink" in the temperature schedule and it appears consistently in multiple runs. Further investigation is needed to better understand this phenomenon and what features of the target distribution or the approximate distribution at t causes the annealing process to slow down. In the high-dimensional Friedman 1 data example, only a few steps are sufficient to accurately capture the unimodal posterior distribution, where a much larger amount of significantly smaller steps are required in the bimodal case, particularly at the beginning, when the inverse temperature t_k is small.

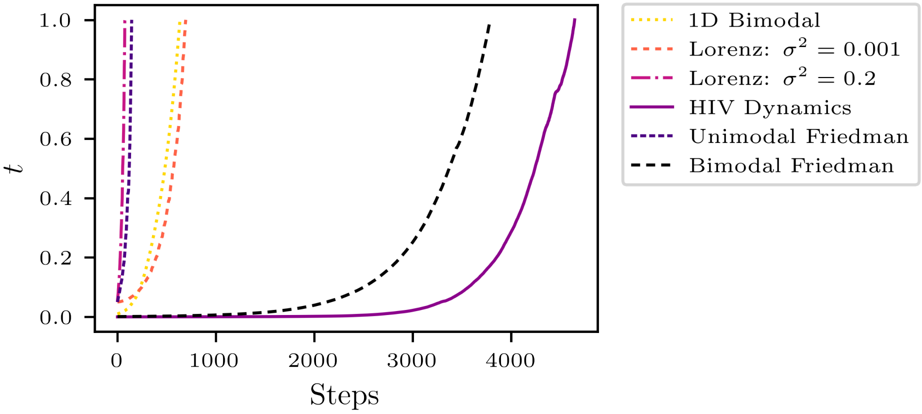


Figure 21: Comparison of annealing schedules for one-dimensional bimodal density (Example 1), the Lorenz attractor (Example 4), the HIV dynamical system (Example 5), and the Friedman 1 dataset (Example 6).

The relevant hyperparameters for AdaAnn (Algorithm 1) in the five examples are summarized in Table 11. One may also want to allow for more gradient updates to be performed for each t_k so that NFs can provide a better approximation of $p^{t_k}(\boldsymbol{Z}, \boldsymbol{X})$, especially for more complex or higher dimensional densities. Except for the one and two dimensional densities,

Table 11: Summary of the AdaAnn hyperparameters used in all 6 examples.

Example	**Description**	τ	t_0	T_0	T	T_1	N	N_1	M
1	1D Bimodal	0.005	0.01	500	2	8,000	100	1,000	1,000
2	1D Parametric Bimodal	0.002	0.01	500	4	8,000	100	1,000	1,000
3	2D Bimodal Density	0.002	0.01	500	3	8,000	100	1,000	1,000
4	Lorenz Attractor	0.2	0.05	500	5	5,000	100	200	100
5	HIV Model	0.005	0.00005	1,000	5	5,000	100	200	100
6	Unimodal Friedman 1	0.2	0.05	500	5	5,000	100	1,000	1,000
6	Bimodal Friedman 1	0.005	0.001	1,000	5	5,000	100	1,000	1,000

we use 5 updates per temperature increase in all of the other examples. At the target temperature of $t = 1$, it is also desirable to perform additional iterations to refine the approximation of the target distribution. For the Lorenz, HIV dynamical system, and Friedman 1 dataset, 5,000 appears to be a reasonable number of iterations leading to an accurate posterior. At $t = 1$, we also typically increase the the batch size.

5 Discussion

We introduced AdaAnn, an adaptive scheduler that automatically suggests changes in the annealing temperature. This scheme has third-order accuracy and is obtained from a Taylor series expansion of the KL divergence between two annealed densities which differ by a sufficiently small inverse temperature increment.

AdaAnn requires two main parameters to be defined: the initial temperature t_0^{-1} and the KL divergence tolerance τ. The choice of t_0 is dependent on the separation and width of the modes in the target distribution. As observed for the HIV dynamical system in Section 4.5, a posterior with very narrow or separated modes requires a smaller t_0, leading to a more uniform initial density. Regarding the KL divergence tolerance, an exceedingly large τ can provide a poor approximation that misses relevant features in the target distributions, e.g., could miss one of the modes in a multimodal posterior. Conversely, a too small τ may result in unnecessary incremental steps and added computational cost yielding no edge in computational efficiency over linear schedulers.

AdaAnn is simple to implement and can lead to significant computational saving compared to *a priori* selected annealing schedules. We demonstrate the application of AdaAnn in planar flows for distribution approximation and variational inference, but no problem is foreseen in applying AdaAnn with other types of flows or other algorithms for the solution of inverse problems (e.g., MCMC).

In future work, we will look into further improving the computational efficiency of AdaAnn. We will also compare AnaAnn with other non-adaptive schedulers besides linear schedulers such as exponential schedulers.

Acknowledgements

All authors gratefully acknowledge the support by the NSF Big Data Science & Engineering grant #1918692 and the computational resources provided through the Center for Research Computing at the University of Notre Dame. DES also acknowledges support from NSF CAREER grant #1942662.

References

[1] E. H. Aarts and J. H. Korst. Boltzmann machines for travelling salesman problems. *European Journal of Operational Research*, 39(1):79–95, 1989.

[2] P. Alquier and J. Ridgway. Concentration of tempered posteriors and of their variational approximations, 2019.

[3] D. J. Bates, J. D. Hauenstein, and N. Meshkat. Identifiability and numerical algebraic geometry. *PLOS ONE*, 14:1–23, 12 2019.

[4] A. Bhattacharya, D. Pati, and Y. Yang. Bayesian fractional posteriors. *The Annals of Statistics*, 47(1):39–66, 2019.

[5] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[6] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, Apr 2017.

[7] L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using real NVP. *arXiv preprint arXiv:1605.08803*, 2016.

[8] J. H. Friedman. Multivariate adaptive regression splines. *The annals of statistics*, 19(1):1–67, 1991.

[9] A. E. Gelfand and A. F. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American statistical association*, 85(410):398–409, 1990.

[10] C. J. Geyer. Markov chain monte carlo maximum likelihood. *Computing Science and Statistics: Proceedings of the 23rd Symposium on the Interface, American Statistical Association. 1991 New York*, pages 156–163, 1991.

[11] R. B. Gramacy. tgp: an r package for bayesian nonstationary, semiparametric nonlinear regression and design by treed gaussian process models. *Journal of Statistical Software*, 19:1–46, 2007.

[12] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[13] C.-W. Huang, S. Tan, A. Lacoste, and A. Courville. Improving explorability in variational inference with annealed variational objectives, 2018.

[14] P. Izmailov, P. Kirichenko, M. Finzi, and A. G. Wilson. Semi-supervised learning with normalizing flows. In *International Conference on Machine Learning*, pages 4615–4630. PMLR, 2020.

[15] M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[16] M. Karabin and S. J. Stuart. Simulated annealing with adaptive cooling rates. *The Journal of Chemical Physics*, 153(11):114103, 2020.

[17] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017.

[18] D. P. Kingma and P. Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039*, 2018.

[19] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29:4743–4751, 2016.

[20] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[21] I. Kobyzev, S. Prince, and M. Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 1–1, 2020.

[22] Z. Kong and K. Chaudhuri. The expressive power of a class of normalizing flow models, 2020.

[23] F. Liang, C. Liu, and R. Carroll. *Advanced Markov chain Monte Carlo methods: learning from past samples.* John Wiley & Sons, 2011.

[24] J. Liu, A. Kumar, J. Ba, J. Kiros, and K. Swersky. Graph normalizing flows. *arXiv preprint arXiv:1905.13177*, 2019.

[25] E. N. Lorenz. Deterministic nonperiodic flow. *Journal of Atmospheric Sciences*, 20(2):130 – 141, 1963.

[26] C. Louizos and M. Welling. Multiplicative normalizing flows for variational bayesian neural networks. In *International Conference on Machine Learning*, pages 2218–2227. PMLR, 2017.

[27] W. Mahdi, S. A. Medjahed, and M. Ouali. Performance analysis of simulated annealing cooling schedules in the context of dense image matching. *Computación y Sistemas*, 21(3):493–501, 2017.

[28] E. Marinari and G. Parisi. Simulated tempering: a new monte carlo scheme. *EPL (Europhysics Letters)*, 19(6):451, 1992.

[29] J. Maroñas, O. Hamelijnck, J. Knoblauch, and T. Damoulas. Transforming gaussian processes with normalizing flows. In *International Conference on Artificial Intelligence and Statistics*, pages 1081–1089. PMLR, 2021.

[30] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

[31] R. M. Neal. Sampling from multimodal distributions using tempered transitions. *Statistics and Computing*, 6(4):353–366, Dec 1996.

[32] R. M. Neal. Slice sampling. *The annals of statistics*, 31(3):705–767, 2003.

[33] G. Papamakarios, T. Pavlakou, and I. Murray. Masked autoregressive flow for density estimation, 2018.

[34] A. Perelson. Modelling viral and immune system dynamics. *Nature reviews. Immunology*, 2:28–36, 02 2002.

[35] R. Ranganath, S. Gerrish, and D. Blei. Black box variational inference. In *Artificial*

Intelligence and Statistics, pages 814–822, 2014.

[36] D. J. Rezende and S. Mohamed. Variational inference with normalizing flows, 2016.

[37] C. P. Robert, G. Casella, and G. Casella. *Monte Carlo statistical methods*, volume 2. Springer, 1999.

[38] S. Strogatz. *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry and Engineering*. Studies in nonlinearity. Westview, 2000.

[39] T. Tieleman and G. Hinton. Lecture 6.5-rmsprop, coursera: Neural networks for machine learning. *University of Toronto, Technical Report*, 2012.

[40] A. Vulpiani, F. Cecconi, and M. Cencini. *Chaos: from simple models to complex systems*, volume 17. World Scientific, 2009.

[41] M. Wainwright and M. Jordan. *Graphical models, exponential families, and variational inference*. Now Publishers Inc, 2008.

[42] Y. Wang, F. Liu, and D. E. Schiavazzi. Variational inference with nofas: Normalizing flow with adaptive surrogate for computationally expensive models. *Journal of Computational Physics*, 467:111454, 2022.

[43] J. Whang, E. Lindgren, and A. Dimakis. Composing normalizing flows for inverse problems. In *International Conference on Machine Learning*, pages 11158–11169. PMLR, 2021.

[44] G. Yang, X. Huang, Z. Hao, M.-Y. Liu, S. Belongie, and B. Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4541–4550, 2019.

A Appendix

A.1 Motivating Bimodal

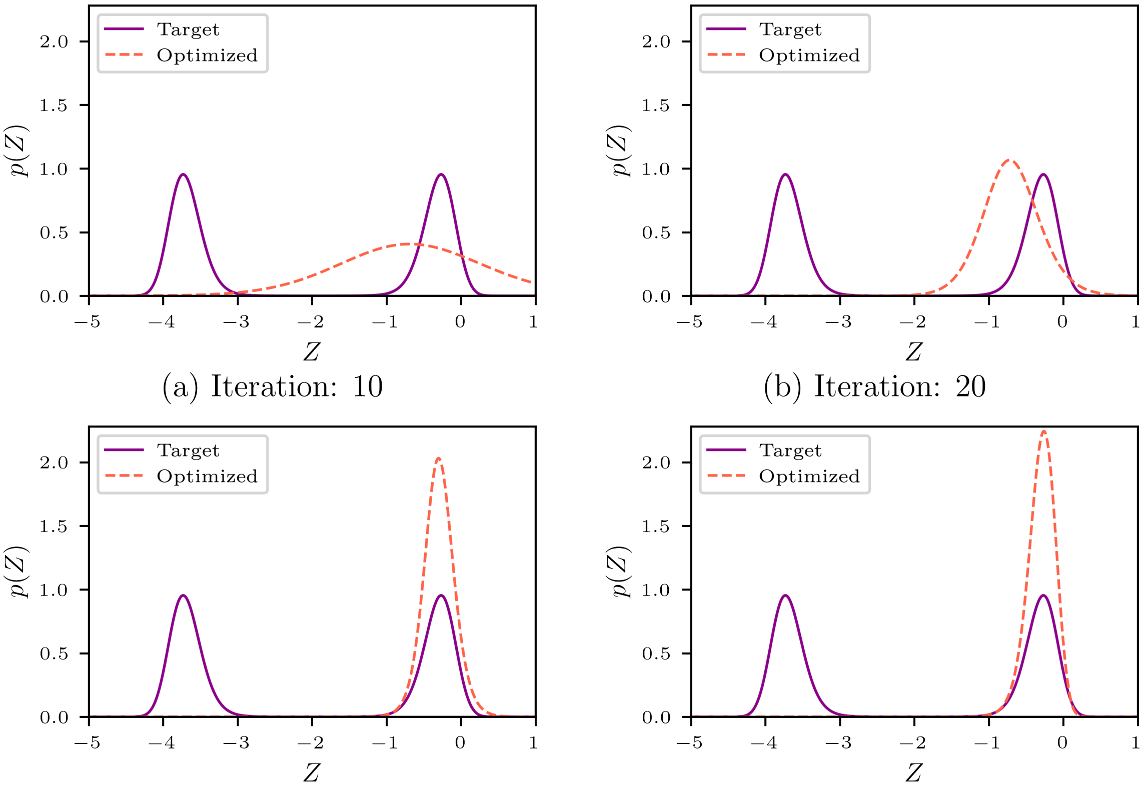
Without using annealing, when the variational distribution moves towards one of the two modes and samples are no longer generated near the other, it is very unlikely that the bimodal character of the density will ever be recovered. This is illustrated in Figure 22.

A.2 1D Case Study

Table 12 provides summarizes results from the asymmetric case of Example 2.

A.3 Lorenz

Figures 23 and 24 compare AdaAnn and linear schuduler for the Lorenz attractor with $\sigma^2 = 0.2$. Tables 13 and 14 compare posterior mean and standard deviation using Monte Carlo integration with values obtained using VI via NFs with AdaAnn and linear schedulers.



(a) Iteration: 10

(b) Iteration: 20

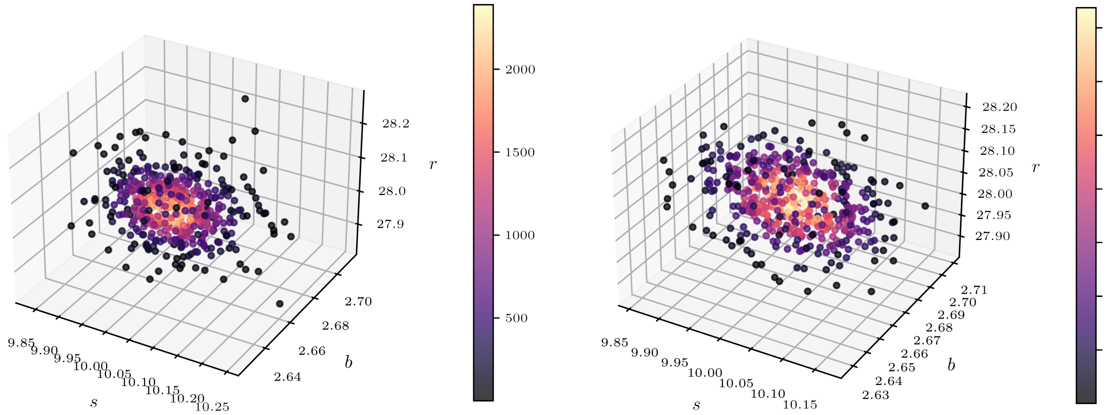
(c) Iteration: 100

(d) Iteration: 8,000

Figure 22: Variational approximation for bimodal density $p(Z)$ without annealing at various iterations.

Table 12: Comparing mean and standard deviation of final loss and total parameter updates for AdaAnn and linear schedulers in the asymmetric case of Example 2.

	Final Loss				**Total Parameter Updates**			
	AdaAnn		**Linear**		**AdaAnn**		**Linear**	
$\boldsymbol{m}$	Mean	SD	Mean	SD	Mean	SD	Mean	SD
1	0.0012	0.0011	0.0007	0.0005	11,463	1,957	17,640	1,643
2	0.0050	0.0037	0.0034	0.0028	11,207	2,299	17,224	2,142
3	0.0298	0.0216	0.0350	0.0354	10,832	2,432	14,964	2,838
4	0.0405	0.0189	0.0361	0.0265	11,561	2,802	14,364	2,675
5	0.0391	0.0145	0.0380	0.0277	9,996	2,557	14,536	2,194
6	0.0415	0.0211	0.0391	0.0425	10,434	2,675	14,996	2,712
7	0.0459	0.0298	0.0398	0.0311	10,606	2,882	14,468	2,909
8	0.0325	0.0147	0.0343	0.0229	10,979	2,815	14,736	2,771



(a) AdaAnn

(b) Linear

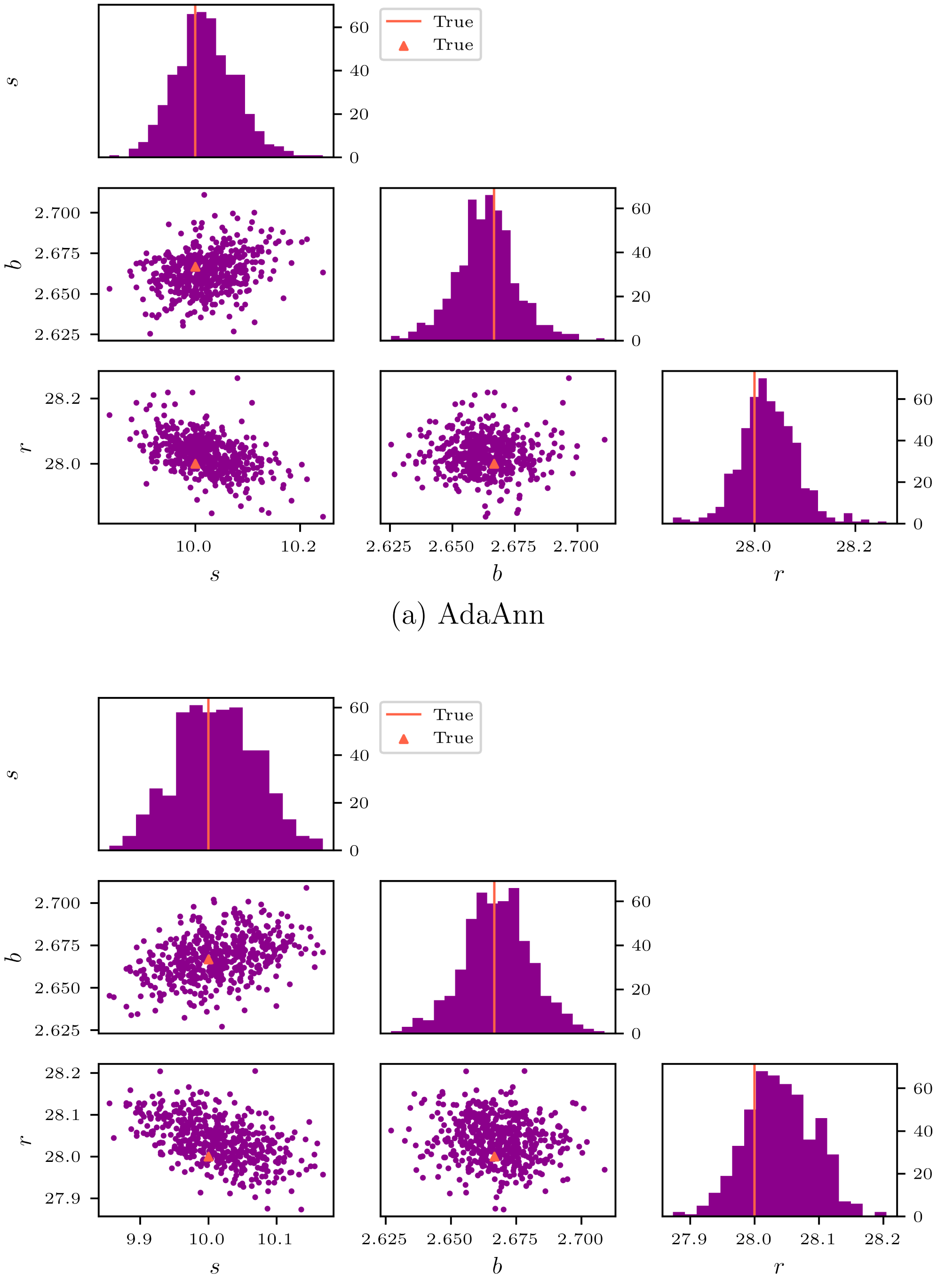
Figure 23: Approximate posterior distribution of parameters (s, b, r) for the Lorenz attractor obtained by VI via NFs with $\sigma^2 = 0.2$ using AdaAnn and linear schedulers.

Table 13: Posterior mean and standard deviation for the parameters of the Lorenz attractor, based on 10,000 samples obtained by VI via NFs with AdaAnn and linear schedules for $\sigma^2 = 0.2$.

True	**MC True**		**AdaAnn**		**Linear**	
Parameter	Mean	SD	Mean	SD	Mean	SD
$s = 10$	10.0152	0.0624	10.0212	0.0632	10.0103	0.0608
$b = 8/3$	2.6668	0.0120	2.6633	0.0118	2.6680	0.0121
$r = 28$	28.0508	0.0556	28.0287	0.0555	28.0366	0.0546

Table 14: Computing mean and standard deviations for posterior distribution with $\sigma^2 = 0.001$ using Monte Carlo integration for varying number of samples.

Sample	$\boldsymbol{s}$		$\boldsymbol{b}$		$\boldsymbol{r}$	
Points	Post. Mean	Post. SD	Post. Mean	Post. SD	Post. Mean	Post. SD
500	10.002752	0.004875	2.666225	0.000832	27.999229	0.003858
1,000	10.002318	0.004163	2.666298	0.000890	27.998961	0.003628
5,000	10.002459	0.004313	2.666234	0.000840	27.998519	0.003649
10,000	10.002827	0.004376	2.666279	0.000819	27.998618	0.003650
20,000	10.002668	0.004350	2.666295	0.000851	27.998585	0.003629
30,000	10.002782	0.004388	2.666294	0.000852	27.998369	0.003740
40,000	10.002782	0.004338	2.666319	0.000855	27.998617	0.003727



(a) AdaAnn

(b) Linear

Figure 24: Marginal posterior distributions and pairwise scatter plots of parameters (s, b, r) for the Lorenz attractor with $\sigma^2 = 0.2$ for AdaAnn and linear schedulers.