Nested Variational Inference

Anonymous Author(s)
Affiliation
Address
email

Abstract

We develop nested variational inference (NVI), a family of methods that learn proposals for nested importance samplers by minimizing an forward or reverse KL divergence at each level of nesting. NVI is applicable to many commonly-used importance sampling strategies and provides a mechanism for learning intermediate densities, which can serve as heuristics to guide the sampler. Our experiments apply NVI to (a) sample from a multimodal distribution using a learned annealing path (b) learn heuristics that approximate the likelihood of future observations in a hidden Markov model and (c) to perform amortized inference in hierarchical deep generative models. We observe that optimizing nested objectives leads to improved sample quality in terms of log average weight and effective sample size.

1 Introduction

Deep generative models provide a mechanism for incorporating priors into methods for unsupervised representation learning. This is particularly useful in settings where the prior defines a meaningful inductive bias that reflects the structure of the underlying domain. Training models with structured priors, however, poses some challenges. A standard strategy for training deep generative models is to maximize a reparameterized variational lower bound with respect to a generative model and an inference model that approximates its posterior [Kingma and Welling, 2013; Rezende et al., 2014]. This approach works well in variational autoencoders with isotropic Gaussian priors, but often fails in models with high-dimensional and correlated latent variables.

In recent years, a range of strategies for improving upon standard reparameterized variational inference have been put forward. These include wake-sleep style variational methods that minimize the forward KL-divergence [Bornschein and Bengio, 2014; Le et al., 2019], as well as sampling schemes that incorporate annealing [Huang et al., 2018], Sequential Monte Carlo [Le et al., 2017; Naesseth et al., 2017; Maddison et al., 2017], Gibbs sampling [Wu et al., 2019; Wang et al., 2018], and MCMC updates [Salimans et al., 2015; Hoffman, 2017; Li et al., 2017]. While these methods offer flexible inference, typically resulting in better approximations to the posterior compared to traditional variational inference methods, they are also model-specific, requiring specialized sampling schemes and gradient estimators, and can not be easily composed with other techniques.

In this paper, we propose nested variational inference, a framework for combining nested importance sampling and variational inference. Nested importance sampling formalizes the construction of proposals by way of nested calls to other importance samplers [Naesseth et al., 2015, 2019], and admits many existing importance sampling strategies as special cases, including methods based on annealing [Neal, 2001] and sequential Monte Carlo [Del Moral et al., 2006]. NVI learns proposals by optimizing a divergence at each level of nesting. Combining nested variational objectives with importance resampling allows us to compute gradient estimates based on incremental weights, which depend only on variables that are sampled locally, rather than on all variables in the model. Doing so yields lower variance weights, whilst maintaining a high sample diversity relative to existing methods.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

2 Background

Stochastic Variational Inference. Stochastic variational methods approximate a target density $\pi(z;\theta) = \gamma(z;\theta)/Z$ using a variational density $q(z;\phi)$ by estimating the gradient of a variational objective. Two common variational objectives are the forward and revserse Kullback-Leibler (KL) divergence, which are both instances of f-divergences,

$$D_f\left(\pi \,\|\, q\right) = \mathop{\mathbb{E}}_{q}\left[f\left(\frac{\pi(z;\theta)}{q(z;\phi)}\right)\right] = \begin{cases} \mathrm{KL}\left(\pi \,\|\, q\right) & f(\omega) = \omega\log\omega, \\ \mathrm{KL}\left(q \,\|\, \pi\right) & f(\omega) = -\log\omega. \end{cases} \tag{1}$$

Stochastic variational inference is commonly combined with maximum likelihood estimation for the parameters θ. We are typically interested in the setting where $\pi(z;\theta)$ is the posterior density $p_\theta(z \,|\, x)$ of a model with latent variables z and observations x. In this setting, $\gamma(z;\theta) = p_\theta(x,z)$ is the joint density of the model, and $Z = p_\theta(x)$ is the marginal likelihood.

In the case of the reverse divergence $\mathrm{KL}\left(q \,\|\, \pi\right)$, also known as the *exclusive* KL divergence, it is common practice to maximize a lower bound $\mathcal{L} = \mathbb{E}_q[\log(\gamma/q)] = \log Z - \mathrm{KL}\left(q \,\|\, \pi\right)$ with respect to θ and ϕ. The gradient of $\mathcal{L}$ can be approximated using reparameterized samples [Kingma and Welling, 2013; Rezende et al., 2014], likelihood-ratio estimators [Wingate and Weber, 2013; Ranganath et al., 2014], or a combination of the two [Schulman et al., 2015; Ritchie et al., 2016a].

In the case of the forward divergence $\mathrm{KL}\left(\pi \,\|\, q\right)$, also known as the *inclusive* KL divergence, stochastic variational methods typically approximate the gradients

$$\frac{\partial}{\partial\theta}\log Z = \mathop{\mathbb{E}}_{\pi}\left[\frac{\partial}{\partial\theta}\log\gamma(z;\theta)\right], \qquad -\frac{\partial}{\partial\phi}\,\mathrm{KL}\left(\pi \,\|\, q\right) = \mathop{\mathbb{E}}_{\pi}\left[\frac{\partial}{\partial\phi}\log q(z;\phi)\right]. \tag{2}$$

This requires samples from π, which itself requires approximate inference. A common strategy, which was popularized in the context of reweighted wake-sleep (RWS) methods [Bornschein and Bengio, 2015; Le et al., 2019], is to use q as a proposal in an importance sampler.

Self-Normalized Importance Samplers. An expectation $\mathbb{E}_\pi[g(z)]$ with respect to π can be rewritten with respect to a proposal density q by introducing an unnormalized importance weight w,

$$\mathop{\mathbb{E}}_{\pi}\left[g(z)\right] = \frac{1}{Z}\mathop{\mathbb{E}}_{q}\left[w\, g(z)\right], \qquad w = \frac{\gamma(z;\theta)}{q(z;\phi)}. \tag{3}$$

Self-normalized estimators use weighted samples $\{w^s, z^s\}_{s=1}^S$ to both approximate the expectation with respect to q, and to compute an estimate $\hat{Z}$ of the normalizer,

$$\mathop{\mathbb{E}}_{\pi}[g(z)] \simeq \hat{g} = \frac{1}{\hat{Z}}\frac{1}{S}\sum_{s=1}^{S} w^s\, g\big(z^s\big), \qquad \hat{Z} = \frac{1}{S}\sum_{s=1}^{S} w^s, \qquad z^s \sim q. \tag{4}$$

This estimator is consistent, i.e. $\hat{g} \xrightarrow{a.s.} \mathbb{E}_\pi[g(z)]$ as the number of samples S increases. However it is biased, since it follows from Jensen's inequality that $1/Z = 1/\,\mathbb{E}_q[\hat{Z}] \leq \mathbb{E}_q[1/\hat{Z}]$. The degree of bias depends on the variance of the importance weight. When $q = \pi$, the importance weight $w = Z$ has 0 variance, and the inequality is tight. In the context of stochastic variational inference, this means that gradient estimates will initially be strongly biased, since there will be a large discrepancy between q and π, but that this bias will decrease as the quality of the approximation improves.

3 Nested Variational Inference

A widely used strategy in importance sampling is to decompose a difficult sampling problem into a series of easier problems. A common approach is to define a sequence of unnormalized densities $\{\gamma_k\}_{k=1}^K$ that interpolate between an initial density $\pi_1 = \gamma_1/Z_1$, for which sampling is easy, and the final target density $\pi_K = \gamma_K/Z_K$. At each step, samples from the preceding density serve to construct proposals for the next density, which is typically combined with importance resampling or application of a Markov chain Monte Carlo (MCMC) operator to improve the average sample quality.

Nested variational inference defines objectives for optimizing importance samplers that target a sequence of densities. To do so, at every step, it minimizes the discrepancy between a *forward density* $\hat{\pi}_k = \hat{\gamma}_k/Z_{k-1}$, which acts as the proposal, and a *reverse density* $\check{\pi}_k = \check{\gamma}_k/Z_k$, which defines an

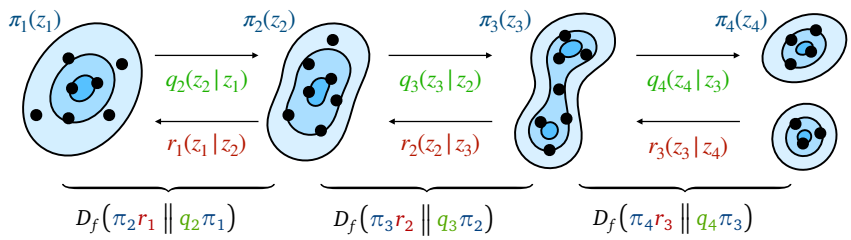

Figure 1: Nested variational inference minimizes an f-divergence at each step in a sequence of densities to learn forward proposals q_k, reverse kernels r_{k-1}, and intermediate densities π_k.

intermediate target. We define the forward density by combining the preceding target γ_{k-1} with a forward kernel q_k, and the reverse density by combining the next target γ_k with a reverse kernel r_{k-1},

$$\check{\gamma}_k(z_k, z_{k-1}) = \gamma_k(z_k)\, r_{k-1}(z_{k-1}|z_k), \qquad \hat{\gamma}_k(z_k, z_{k-1}) = q_k(z_k|z_{k-1})\, \gamma_{k-1}(z_{k-1}). \tag{5}$$

Our goal is to learn pairs of densities $\check{\pi}_k$ and $\hat{\pi}_k$ that are as similar as possible. To do so, we minimize a variational objective $\mathcal{D}$ that comprises an f-divergence for each step in the sequence, along with a divergence between the first intermediate target π_1 and an initial proposal q_1,

$$\mathcal{D} = D_f\left(\pi_1 \,\middle|\middle|\, q_1\right) + \sum_{k=2}^{K} D_f\left(\check{\pi}_k \,\middle|\middle|\, \hat{\pi}_k\right). \tag{6}$$

We can optimize this objective in two ways. The first is to optimize with respect to q_1 and the forward densities $\hat{\pi}_k$. This serves to learn proposals q_k, but can also be used to learn intermediate densities π_k that are as similar as possible to the next density π_{k+1}. In settings where we wish to learn reverse kernels r_k, we can additionally minimize $\mathcal{D}$ with respect to the reverse densities $\check{\pi}_k$. Since each intermediate density π_k occurs in both $\check{\pi}_k$ and in $\hat{\pi}_{k+1}$, this defines a trade-off between maximizing the similarity to π_{k+1} and the similarity to π_{k-1}.

3.1 Nested Importance Sampling and Proper Weighting

Sampling from a sequence of densities can be performed using a nested importance sampling construction [Naesseth et al., 2015, 2019], which uses weighted samples from π_{k-1} as proposals for π_k. The technical requirements for such constructions can be summarized as follows:

Definition 3.1 (Proper weighting)**.** *Let π be a probability density. For some constant $c > 0$, a random pair $(w, z) \sim \Pi$ is properly weighted (p.w.) for an unnormalized probability density $\gamma \equiv Z\pi$ if $w \geq 0$ and for all measurable functions g it holds that*

$$\mathop{\mathbb{E}}_{z,w\sim\Pi}\left[w\, g(z)\right] = c \int dz\, \gamma(z)\, g(z) = cZ \mathop{\mathbb{E}}_{z\sim\pi}\left[g(z)\right].$$

Given a pair (w_{k-1}, z_{k-1}) that is properly weighted for γ_{k-1}, we can use a sequential importance sampling construction to define a pair (w_k, z_k) that is properly weighted for γ_k,

$$z_k \sim q_k(\cdot \mid z_{k-1}), \qquad w_k = v_k\, w_{k-1}, \qquad v_k = \frac{\check{\gamma}_k(z_k, z_{k-1})}{\hat{\gamma}_k(z_k, z_{k-1})}. \tag{7}$$

We refer to the ratio v_k as the incremental weight. In this construction, it is easy to see that (w_k, z_k) is properly weighted for $\check{\gamma}_k$ which implies that (w_k, z_k) is also properly weighted for γ_k, since

$$\int dz_{k-1}\, \check{\gamma}_k(z_k, z_{k-1}) = \int dz_{k-1}\, \gamma_k(z_k) r_{k-1}(z_{k-1} \mid z_k) = \gamma_k(z_k). \tag{8}$$

This construction can be composed with operations that preserve proper weighting, including rejection sampling, application of an MCMC transition operator, and importance resampling. This defines a class of samplers that admits a number of popular methods as special cases, including sequential Monte Carlo (SMC) [Doucet et al., 2001], annealed importance sampling (AIS) [Neal, 2001], and SMC samplers [Chopin, 2002]. These samplers differ in the sequences of densities they define. SMC for state-space models employs the filtering distribution on the first k points in a time series as the intermediate target π_k. In this setting, where the dimensionality of the support increases at each step, we typically omit the reverse kernel r_{k-1}. In AIS and SMC samplers, where the support is fixed, a common strategy is to define an annealing path $\gamma_k(z_k) = \gamma_K(z_k)^{\beta_k}\gamma_1(z_k)^{1-\beta_k}$ that interpolates between the initial density γ_1 and the final target γ_K by varying the coefficients β_k from 0 to 1.

3.2 Computing Gradient Estimates

The NVI objective can be optimized with respect to three sets of densities. We will use θ_k, $\hat{\phi}_k$, and $\check{\phi}_k$ to denote the parameters of the densities π_k, q_k, and r_k respectively. For notational convenience, we use $\hat{\rho}_k = \{\hat{\phi}_k, \theta_{k-1}\}$ to refer to the parameters of the forward density $\hat{\pi}_k$, and $\check{\rho}_k = \{\theta_k, \check{\phi}_{k-1}\}$ to refer to the parameters of the reverse density $\check{\pi}_k$.

Gradients of the Forward KL divergence. When we employ the forward KL as the objective, the derivative with respect to $\hat{\rho}_k$ can be expressed as (see Appendix D.3),

$$-\frac{\partial}{\partial \hat{\rho}_k} \mathrm{KL}\left(\check{\pi}_k \,\|\, \hat{\pi}_k\right) = \mathop{\mathbb{E}}_{\check{\pi}_k}\left[\frac{\partial}{\partial \hat{\rho}_k} \log \hat{\gamma}_k\left(z_k, z_{k-1}; \hat{\rho}_k\right)\right] - \mathop{\mathbb{E}}_{\pi_{k-1}}\left[\frac{\partial}{\partial \hat{\rho}_k} \log \gamma_{k-1}\left(z_{k-1}; \theta_{k-1}\right)\right]. \tag{9}$$

This case is the nested analogue of RWS-style variational inference. We can move the derivative into the expectation, since $\check{\pi}_k$ does not depend on $\hat{\rho}_k$. We then decompose $\log \hat{\pi}_k = \log \hat{\gamma}_k - \log Z_{k-1}$ and use the identity from Equation 2 to express the gradient $\log Z_{k-1}$ as an expectation with respect to π_{k-1}. The resulting expectations can be approximated using self-normalized estimators based on the outgoing weights w_k and incoming weights w_{k-1} respectively.

The gradient of the forward KL with respect to $\check{\rho}_k$ is more difficult to approximate, since the expectation is computed with respect to $\check{\pi}_k$, which depends on the parameters $\check{\rho}_k$. The gradient of this expectation has the form (see Appendix D.3)

$$\begin{aligned} -\frac{\partial}{\partial \check{\rho}_k} \mathrm{KL}\left(\check{\pi}_k \,\|\, \hat{\pi}_k\right) &= -\mathop{\mathbb{E}}_{\check{\pi}_k}\left[\log v_k \, \frac{\partial}{\partial \check{\rho}_k} \log \check{\pi}_k\left(z_k, z_{k-1}; \check{\rho}_k\right)\right], \\ &= -\mathop{\mathbb{E}}_{\check{\pi}_k}\left[\log v_k \, \frac{\partial}{\partial \check{\rho}_k} \log \check{\gamma}_k\left(z_k, z_{k-1}; \check{\rho}_k\right)\right] + \mathop{\mathbb{E}}_{\check{\pi}_k}\left[\log v_k\right] \mathop{\mathbb{E}}_{\pi_k}\left[\frac{\partial}{\partial \check{\rho}_k} \log \gamma_k\left(z_k; \theta_k\right)\right]. \end{aligned} \tag{10}$$

In principle we can approximate this gradient using self-normalized estimators based on the outgoing weight w_k. However, in preliminary experiments we found these estimators to be unstable, particularly for the gradient with respect to the parameters of the reverse kernel r_{k-1}. In this estimator we *decrease* the probability of high-weight samples and *increase* the probability of low-weight samples, rather than the other way around. We hypothesize this leads to problems during early stages of training, when the estimator will underrepresent low-weight samples, for which the probability should increase. For this reason, we employ the reverse KL when learning reverse kernels.

Gradients of the Reverse KL divergence. When computing the gradient of the reverse KL with respect to $\hat{\rho}_k$, we obtain the nested analogue of methods that maximize a lower bound. Here we can either use reparameterized samples [Kingma and Welling, 2013; Rezende et al., 2014] or likelihood-ratio estimators [Wingate and Weber, 2013; Ranganath et al., 2014]. We will follow Ritchie et al. [2016b] and define a unified estimator in which proposals are generated using a construction

$$w_k = v_k w_{k-1}, \qquad z_k = g(\tilde{z}_k, \hat{\phi}_k), \qquad \tilde{z}_k \sim \tilde{q}_k(\tilde{z}_k \mid z_{k-1}, \hat{\phi}_k), \qquad w_{k-1}, z_{k-1} \sim \Pi_{k-1}. \tag{11}$$

This construction recovers reparameterized samplers in the special case when $\tilde{q}_k$ does not depend on parameters, and recovers non-reparameterized samplers when $z_k = \tilde{z}_k$. This means it is applicable to models with continuous variables, discrete variables, or a combination of the two. The gradient of the reverse KL for proposals that are constructed in this manner becomes (see Appendix D.2)

$$\begin{aligned} -\frac{\partial}{\partial \hat{\rho}_k} \mathrm{KL}\left(\hat{\pi}_k \,\|\, \check{\pi}_k\right) = \mathop{\mathbb{E}}_{\check{\pi}_k}\left[\frac{\partial}{\partial z_k} \log \hat{\gamma}_k\left(z_k, z_{k-1}; \hat{\rho}_k\right) \frac{\partial z_k}{\partial \hat{\rho}_k}\right] + \mathop{\mathbb{E}}_{\check{\pi}_k}\left[\log v_k \frac{\partial}{\partial \hat{\rho}_k} \log \hat{\gamma}_k\left(z_k, z_{k-1}; \hat{\rho}_k\right)\right] \\ - \mathop{\mathbb{E}}_{\check{\pi}_k}\left[\log v_k\right] \mathop{\mathbb{E}}_{\pi_{k-1}}\left[\frac{\partial}{\partial \hat{\rho}_k} \log \gamma_{k-1}\left(z_{k-1}; \theta_{k-1}\right)\right]. \end{aligned} \tag{12}$$

In this gradient, the first term represents the pathwise derivative with respect to reparameterized samples. The second term defines a likelihood-ratio estimator in terms of the unnormalized density $\hat{\gamma}_k$, and the third term computes the contribution of the gradient of the log normalizer $\log Z_{k-1}$.

Computing the gradient of the reverse KL with respect to $\check{\rho}_k$ is once again straightforward, since we are computing an expectation with respect to $\hat{\pi}_k$, which does not depend on $\check{\rho}_k$. This means we can move the derivative into the expectation, which yields a gradient analogous to that in Equation 9,

$$-\frac{\partial}{\partial \check{\rho}_k} \mathrm{KL}\left(\hat{\pi}_k \,\|\, \check{\pi}_k\right) = \mathop{\mathbb{E}}_{\hat{\pi}_k}\left[\frac{\partial}{\partial \check{\rho}_k} \log \check{\gamma}_k\left(z_k, z_{k-1}; \check{\rho}_k\right)\right] - \mathop{\mathbb{E}}_{\pi_k}\left[\frac{\partial}{\partial \check{\rho}_k} \log \gamma_k\left(z_k; \theta_k\right)\right]. \tag{13}$$

Variance Reduction. To reduce the variance of the gradient estimates we use the expected log-incremental weight as a baseline for the score function terms and employ the sticking-the-landing trick [Roeder et al., 2017] when reparameterizing the forward kernel as described in Appendix D.

3.3 Relationship to Importance-Weighted and Self-Normalized Estimators

There exists a large body of work on methods that combine variational inference with MCMC and importance sampling. We refer to Appendix A for a comprehensive discussion of related and indirectly related approaches. To position NVI in the context of the most directly related work, we here focus on its relationship to commonly used importance-weighted and self-normalized estimators.

NVI differs from existing methods in that it defines an objective pairs of variables (z_k, z_{k-1}) at each level of nesting, rather than a single objective for the entire sequence of variables $(z_1, \dots, z_K)$. One of the standard approaches for combining importance sampling and variational inference is to define an "importance-weighted" stochastic lower bound $\hat{\mathcal{L}}_K$ [Burda et al., 2016],

$$\hat{\mathcal{L}}_K = \log \hat{Z}_K, \qquad \hat{Z}_K = \frac{1}{S} \sum_{s=1}^{S} w_K^s.$$

By Jensen's inequality, $\mathbb{E}[\hat{\mathcal{L}}_K] \leq \log \mathbb{E}[\hat{Z}_K] = \log Z_K$, which implies that we can define a stochastic lower bound using any properly-weighted sampler for γ_K, including samplers based on SMC [Le et al., 2018; Naesseth et al., 2018; Maddison et al., 2017]. For purposes of learning the target density γ_K, this approach is equivalent to computing an RWS-style estimator of the gradient in Equation 2,

$$\frac{\partial}{\partial \theta_K} \hat{\mathcal{L}}_K = \frac{1}{\hat{Z}_K} \frac{1}{S} \sum_{s=1}^{S} w_K^s \frac{\partial}{\partial \theta_K} \log \gamma_K(z_K^s; \theta_K). \tag{14}$$

However, these two approaches are not equivalent for purposes of learning the proposals. We can maximize a stochastic lower bound to learn q_k, but this requires doubly-reparameterized estimators [Tucker et al., 2018] in order to avoid problems with the signal-to-noise ratio in this estimator, which can paradoxically deteriorate with the number of samples [Rainforth et al., 2018]. The estimators in NVI do not suffer from this problem, since we do not compute the logarithm of an average weight.

NVI is also not equivalent to learning proposals with RWS-style estimators. If we use sequential importance sampling (SIS) to generate samples, a self-normalized gradient for the parameters of q_k that is analogous to the one in Equation 2 has the form

$$\mathop{\mathbb{E}}_{\pi_K, r_{K-1}, \dots, r_1} \left[\frac{\partial}{\partial \hat{\phi}_k} \log q_k(z_k \mid z_{k-1}\,;\, \hat{\phi}_k) \right] \simeq \frac{1}{\hat{Z}_K} \frac{1}{S} \sum_{s=1}^{S} w_K^s \frac{\partial}{\partial \hat{\phi}_k} \log q_k(z_k^s \mid z_{k-1}^s\,;\, \hat{\phi}_k). \tag{15}$$

Note that this expression depends on the final weight w_K. By contrast, a NVI objective based on the forward KL yields a self-normalized estimator that is defined in terms of the intermediate weight w_k

$$\mathop{\mathbb{E}}_{\pi_k, r_{k-1}} \left[\frac{\partial}{\partial \hat{\phi}_k} \log q_k(z_k \mid z_{k-1}\,;\, \hat{\phi}_k) \right] \simeq \frac{1}{\hat{Z}_k} \frac{1}{S} \sum_{s=1}^{S} w_k^s \frac{\partial}{\partial \hat{\phi}_k} \log q_k(z_k^s \mid z_{k-1}^s\,;\, \hat{\phi}_k). \tag{16}$$

If instead of SIS we employ sequential importance resampling (i.e. SMC), then the incoming weight w_{k-1} is identical for all samples. This means that we can express this estimator in terms of the incremental weight v_k rather than the intermediate weight w_k

$$\mathop{\mathbb{E}}_{\pi_k, r_{k-1}} \left[\frac{\partial}{\partial \hat{\phi}_k} \log q_k(z_k \mid z_{k-1}\,;\, \hat{\phi}_k) \right] \simeq \sum_{s=1}^{S} \frac{v_k^s}{\sum_{s'=1}^{S} v_k^{s'}} \frac{\partial}{\partial \hat{\phi}_k} \log q_k(z_k^s \mid z_{k-1}^s\,;\, \hat{\phi}_k). \tag{17}$$

We see that NVI allows us to compute gradient estimates that are localized to a specific level of the sampler. In practice, this can lead to lower-variance gradient estimates.

Having localized gradient computations also offers potential memory advantages. Existing methods typically perform reverse-mode automatic differentiation on an objective that is computed from the final weights (e.g. the stochastic lower bound). This means that memory requirements scale as $\mathcal{O}(SK)$ since the system needs to keep the entire computation graph in memory. In NVI, gradient estimates at level k do not require differentiation of the incoming weights w_{k-1}, This means that it is possible to perform automatic differentiation on a locally-defined objective before proceeding to the next level of nesting, which means that memory requirements would scale as $\mathcal{O}(S)$. It should therefore in principle be possible to employ a large number of levels of nesting K in NVI, although we do not evaluate the stability of NVI at large K in our experiments.

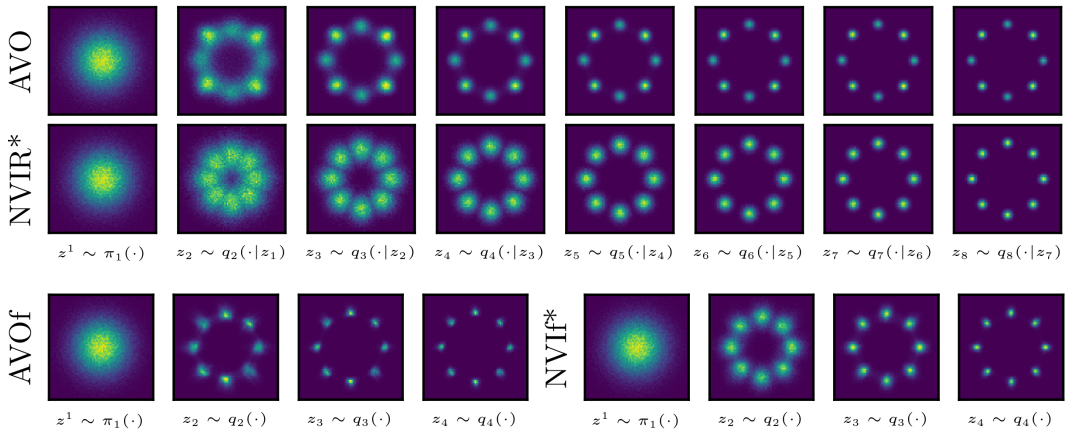

Figure 2: (*Top*) Samples from forward kernels trained with AVO, and NVIR*. (*Bottom*) Samples from flow-based proposals trained with AVOf, and NVIRf*.

4 Experiments

We evaluate NVI on three tasks, (1) learning to sample form an unnormalized target density where intermediate densities are generated using annealing, (2) learning heuristic factors to approximate the marginal likelihood of future observations in state-space models, and finally (3) inferring distributions over classes from small numbers of examples in deep generative Bayesian mixtures.

4.1 Sampling from Multimodal Densities via Annealing

A common strategy when sampling from densities with multiple isolated modes is to anneal from an initial density γ_1, which is typically a unimodal distribution that we can sample from easily, to the target density γ_K, which is multimodal [Neal, 2001]. Recent work on annealed variational objectives (AVOs) learns forward kernels q_k and reverse kernels r_{k-1} for an annealing sequence by optimizing a variational lower bound at each level of nesting [Huang et al., 2018], which is equivalent to minimizing an NVI based on the reverse KL for a fixed sequence of densities γ_k,

$$\max_{q_k, r_k} \mathcal{L}_k^{\text{AVO}}, \quad \mathcal{L}_k^{\text{AVO}} = \mathop{\mathbb{E}}_{q_1,\ldots,q_k}\big[\log v_k\big], \quad \gamma_k(z) = q_1(z)^{1-\beta_k}\gamma_K(z)^{\beta_k}, \quad k = 1, \ldots, K. \tag{18}$$

NVI allows us to improve upon AVO in two ways. First, we can perform importance resampling at every step to optimize an SMC sampler rather than an annealed importance sampler. Second, we learn annealing paths $(\beta_1, \ldots, \beta_K)$ which schedule intermediate densities γ_k such that the individual KL divergences, and hence the expected log incremental importance weights, are minimized.

We illustrate the effect of these two modifications in Figure 2, in which we compare AVO to NVI with resampling and a learned path, which we refer to as NVIR*. Both methods minimize the reverse KL at each step [1]. For details on network architectures see Appendix E.1. The learned annealing path in NVIR* results in a smoother interpolation between the initial and final density. We also see that AVO does not assign equal mass to all 8 modes, whereas NVIR* yields a more even distribution.

In Figure 3 we compute the reverse KL between targets at each step in the sequence. For the standard linear annealing path, the KL decreases with k, suggesting that later intermediate densities are increasingly redundant. By constrast, in NVIR* we see that the KL is approximately constant across the path, which is what we would expect when minimizing $\mathcal{D}$ with respect to β_k. This is also the case in an ablation without resampling, which we refer to as NVI*.

Figure 4 shows a rolling average of the ESS and its variance during training. We compare NVI-based methods to SVI and a variational SMC sampler [Le et al., 2017; Maddison et al., 2017; Naesseth et al., 2017]. NVIR* has consistently higher ESS and significantly lower variance compared to baselines. These plots also provides insight into the role of resampling in training dynamics. In NVI*, we observe a cascading convergence, which is absent in NVIR*. We hypothesize that resampling reduces the reliance on high-quality proposals from step $k-1$ when estimating gradients at step k.

[1] As noted in Section 3.2, we found optimization of the forward KL to be unstable when learning r_k.

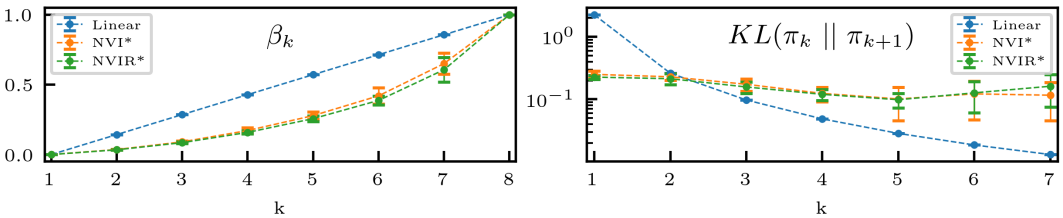

Figure 3: (*Left*) Annealing paths learned by NVI* and NIVR* and the linearly spaced annealing geometric schedule (Linear) used by AVO, NVI, and NVIR. Results are averaged over 10 restarts; error bars indicate two standard deviations. (*Right*) The KL-divergences (computed by numeric integration) between consecutive intermediate distributions for different schedules.

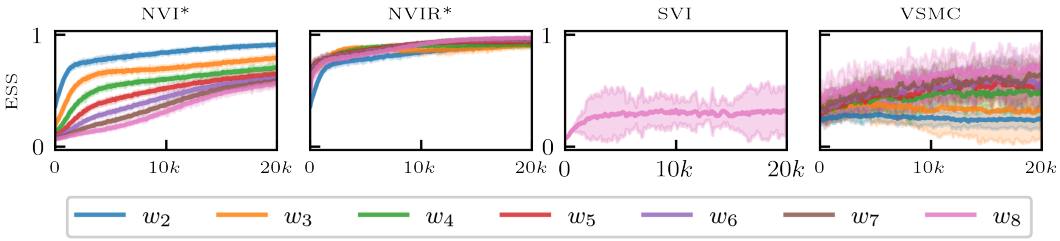

Figure 4: ESS during training for different methods using 7 pairs of transition kernels (sequence length K=8) averaged across 10 independent runs. Error bars indicate ± 2 standard deviations; mean and standard deviation are computed based a rolling average with window size 10.

	$\log \hat{\mathbf{Z}}$		($\log Z \approx 2.08$)		**ESS**			
Seq. length	K=2	K=4	K=6	K=8	K=2	K=4	K=6	K=8
SVI	1.86	1.89	1.92	1.72	51	47	32	25
SVI-flow	2.06	-	-	-	55	-	-	-
AVO	1.86	1.96	2.01	2.05	51	44	46	46
NVI	1.86	1.97	2.03	2.06	51	45	45	41
NVIR	1.86	1.98	2.04	2.06	51	**99**	**98**	**97**
NVI*	1.86	**2.06**	**2.07**	2.07	51	51	54	54
NVIR*	1.86	**2.06**	**2.07**	**2.08**	51	95	96	**97**
AVO-flow	2.05	2.08	2.07	**2.08**	28	67	77	**70**
NVI*-flow	2.05	**2.08**	**2.08**	**2.08**	28	**81**	**78**	**70**

Table 1: Sample efficiency for NVI variants and baselines for $K-1$ annealing steps and L samples per step for a fixed budget of $K \cdot L = 288$ samples. Metrics are computed for 100 batches of 100 samples per model across 10 restarts.

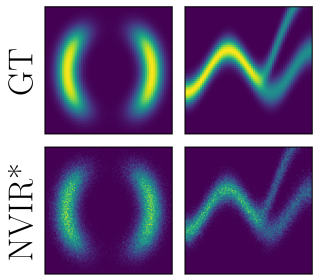

Figure 5: Ground truth densities (GT) and samples from final target density for NVIR* with 2 intermediate densities.

Annealed NVI has similar use cases as normalizing flows Rezende and Mohamed [2015]. Inspired by concurrent work of Arbel et al. [2021], which explores a similar combination of SMCS and normalizing flows, we additionally compare flow-based versions of NVI to planar normalizing flows, which maximizing a standard lower bound (SVI-flow). We found that a normalizing flows can be effectively trained with NVI, in that samplers produce better estimates of the normalizing constant and higher ESS compared to SVI-flow. We also find that flow based models are able to produce high quality samples with fewer intermediate densities (Figure 2, bottom). Moreover, we see that combining a flow-based proposal with learned β_k values (NVI*-flow) results in a more accurate approximation of the target than in an ablation with a linear interpolation path (AVO-flow)[2], resulting in a poor approximation to the target density (see Appendix E.1).

In Table 1 we report sample quality in terms of the stochastic lower bound $\hat{\mathcal{L}}_K = \log \hat{Z}_K$ and the effective sample size ESS $= \sum_s (w_K^s)/(\sum_s w_K^s)^2$. The first metric can be interpreted as a measure of the average sample quality, whereas the second metric quantifies sample diversity. We compare NVI with and without resampling (NVIR* and its NVI*) to ablations with a linear annealing path (NVIR and NVI), AVO, and a standard SVI baseline in which there are no intermediate densities. We additionally compare against AVO-flow and NVI*-flow, which employ flows. We observe that

[2] AVO-flow is not technically a baseline, but an (incrementally) novel combination of AVO and flows.

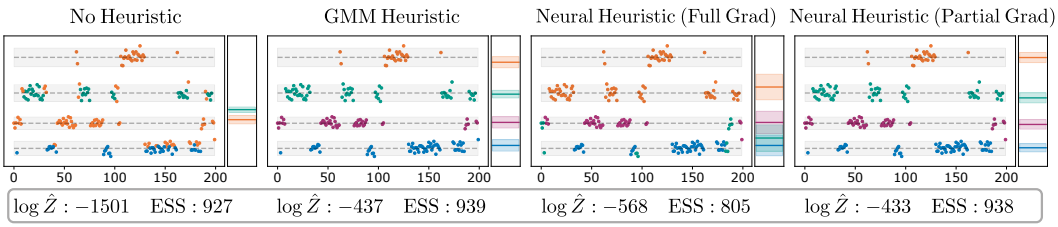

Figure 6: (*Top*) qualitative results of 1 instance with $K = 200$ time steps (x-axis). Observations are color-coded based on the inferred assignments; Each colored band corresponds to the inferred cluster mean with one standard deviation, where the grey bands are the ground truth of the clusters. (*Bottom*) We compute $\log \hat{Z}$ and ESS using 1000 samples and report average values on 2000 test instances.

NVIR* and NVI*-flow outperform ablations and baselines in terms of $\log \hat{Z}$, and are competitive in terms of ESS. We show qualitative results for two additional target densities in Figure 5.

4.2 Learning Heuristic Factors for state-space models

Sequential Monte Carlo methods are commonly used in state-space models to generate samples by proposing one variable at a time. To do so, they define a sequence of densities $\pi_k = \gamma_k / Z_k$ on the first k time points in a model, which are also known as the filtering distributions,

$$\gamma_k(z_{1:k}, \eta) = p(x_{1:k}, z_{1:k}, \eta) = p(\eta)\, p(x_1, z_1 \,|\, \eta) \prod_{l=2}^{k} p(x_l, z_l \,|\, z_{l-1}, \eta), \quad Z_k = p(x_{1:k}). \tag{19}$$

Here $z_{1:k}$ and $x_{1:k}$ are sequences of hidden states and observations, and η is a set of global variables of the model. These densities γ_k differ from those in the annealing task in Section 4.1 in that the dimensionality of the support increases at each step, whereas all densities had the same support in the previous experiment. In this context, we can define a forward density $\hat{\gamma}_k$ that combines the preceding target γ_{k-1} with a proposal q_k for the time point z_k, and define a reverse density $\check{\gamma}_k = \gamma_k$ that is equal to the next intermediate density (which means that we omit r_{k-1}),

$$\check{\gamma}_k(z_{1:k}, \eta) = \gamma_k(z_{1:k}, \eta), \qquad \hat{\gamma}_k(z_{1:k}, \eta) = q_k(z_k \,|\, z_{1:k-1})\, \gamma_{k-1}(z_{1:k-1}, \eta). \tag{20}$$

A limitation of this construction is that the filtering distribution π_{k-1} is not always a good proposal, since it does not incorporate knowledge of future observations. Ideally, we would like to define intermediate densities $\gamma_k(z_{1:k}, \eta) = p(x_{1:K}, z_{1:k}, \eta)$ that correspond to the smoothing distribution, but this requires computation of the marginal likelihood of future observations $p(x_{k+1:K} \,|\, z_k, \eta)$, which is intractable. This is particularly problematic when sampling η as part of the SMC construction. The first density $\pi_1(z_1, \eta) = p(z_1, \eta \,|\, x_1)$ will be similar to the prior, which will result in poor sampler efficiency, since the smoothing distribution $p(z_1, \eta \,|\, x_{1:K})$ will typically be much more concentrated.

To overcome this problem, we will use NVI to learn heuristic factors ψ_θ that approximate the marginal likelihood of future observations. We define a sequence of densities $(\gamma_0, \ldots, \gamma_K)$,

$$\gamma_0(\eta) = p(\eta)\, \psi_\theta(x_{1:K} | \eta), \quad \gamma_k(z_{1:k}, \eta) = p(x_{1:k}, z_{1:k}, \eta)\, \psi_\theta(x_{k+1:K} \mid \eta), \quad k = 1, 2, ..., K.$$

Our goal is to learn parameters θ of the heuristic factor to ensure that intermediate densities approximate the smoothing distribution. This approach is similar to recently proposed work on twisted variational SMC Lawson et al. [2018], which maximized a stochastic lower bound.

To evaluate the this approach, we will learn heuristic factors for a hidden Markov model (HMM). While HMMs are a well-understood model class, they are a good test case, in that they give rise to significant sample degeneracy in SMC and allow computation of tractable heuristics. We optimize an NVI objective based on the forward KL with respect to the heuristic factor ψ_θ, an initial proposal $q_\phi(\eta \,|\, x_{1:K})$ and a forward kernel $q_\phi(z_k \,|\, x_k, z_{k-1}, \eta)$. We train on 10000 simulated HMM instances, each containing $K = 200$ time steps and $M = 4$ states. For network architectures see Appendix E.2.

Figure 6 shows qualitative and quantitative results. We compare partial optimization with respect to $\hat{\gamma}_k$ to full optimization with respect to both $\hat{\gamma}_k$ and $\check{\gamma}_k$. We also compare NVI with neural heuristics to a baseline without heuristics, and a baseline that uses a Gaussian mixture model as a hand-coded heuristic. While full optimization yields poor results, partial optimization learns a neural heuristic whose performance is similar to the GMM heuristic, which is a strong baseline in this context.

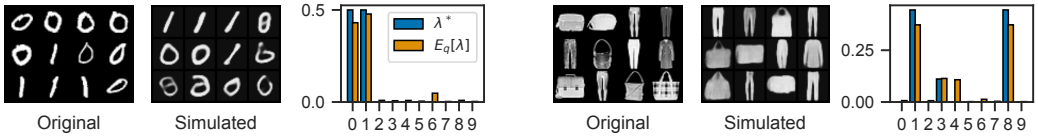

Figure 7: BGMM-VAE trained on MNIST & FashionMNIST. (*Left*) Samples from a test mini-batch of size $N = 300$. (*Middle*) Samples from the generative model, generated from the λ inferred from the test mini-batch. (*Right*) Comparison of ground truth λ^* and the expected inferred value.

4.3 Meta Learning with Deep Generative models

In this experiment, we evaluate NVI in the context of deep generative models with hierarchically-structured priors. Concretely, we consider the task of inferring class weights from a mini-batch of images in a fully unsupervised manner. For this purpose, we employ a variational autoencoders (VAE) with a prior in the form of a Bayesian Gaussian mixture model. We evaluate our model based on the quality of both the generative and inference model.

Model Description. We define a hierarchical deep generative model for batches of N images of the form (see Appendix E.3 for a graphical model and architecture description)

$$\lambda \sim \text{Dir}(\cdot\,;\,\alpha) \quad c_n \sim \text{Cat}(\cdot\,|\,\lambda) \quad z_n \sim \mathcal{N}(\cdot\,|\,\mu_{c_n}, 1/\tau_{c_n}) \quad x_n \sim p(\cdot\,|\,z_n;\theta) \quad \text{for } n = 1 \dots N.$$

Here λ, c_n, z_n, and x_n refer to the cluster probabilities, cluster assignments, latent codes, and observed images respectively. In the likelihood $p(x|z;\theta_x)$, we use a convolutional network to parameterize a continuous Bernoulli distribution [Loaiza-Ganem and Cunningham, 2019]. We define proposals $q(z|x;\phi_z)$, $q(c|z;\phi_c)$, and $q(\lambda|c;\phi_\lambda)$, which are also parameterized by neural networks. We refer to this model as Bayesian Gaussian mixture model VAE (BGMM-VAE).

Objective. To construct an NVI objective, we define intermediate densities for c and z. Unlike in previous experiments, we employ tractable densities in the form of a categorical $\pi_c(c;\theta_c)$ for cluster assignments and a 8-layer planar flow $\pi_z(z;\theta_z)$ for the latent codes. We minimize the forward KL for the first two levels of nesting and the reverse KL at the final level of nesting

$$\text{KL}\left(p(\lambda)\,p(c|\lambda)\middle\|\pi_c(c)\,q(\lambda|c)\right)+\text{KL}\left(\pi_c(c)\,p(z|c)\middle\|\pi_z(z)\,q(c|z)\right)+\text{KL}\left(\hat{p}(x)\,q(z|x)\middle\|\pi_z(z)\,p(x|z)\right),$$

where $\hat{p}(x)$ is an empirical distribution over mini-batches of training data. We optimize the first two terms with respect to π and q, and the third term with respect q only. Since the intermediate densities are tractable in this model, no nested construction is required; we can compute gradient estimates base on a (single) samples from $p(\lambda)p(c|\lambda)$ in the first term, $\pi_c(c)q(\lambda|c)$ in the second, and $\hat{p}(x)q(z|x)$ in the final term. To learn the parameters $\{\mu, \tau, \theta_x\}$ of the generative model, we maximize a single-sample approximation of a lower bound $\mathcal{L} = \mathbb{E}_q\left[\log\left(p(x,z,c,\lambda)\,/\,q(z,c,\lambda|x)\right)\right]$.

Results. We evaluate NVI for the BGMM-VAE using the following procedure. We generate mini-batches with a sampled λ^* (for which we make use of class labels that are not provided to the model). We then compute the expectation of λ under $q(\lambda, c, z|x)$ by sampling from the inference model, and compare this value against λ^*. Additionally, we generate a new mini-batch given the inferred λ by running the generative model forward. The results are shown in Figure 7. The cluster indices are rearranged based on the association of clusters to true classes. We observe that both the inferred λ and the generated samples match the test mini-batch reasonably well. When we train with RWS, however with the same setting, the posterior approximation is quite poor (Figure 9) which we hypothesise that is due to relying on the gradients of joints rather than local gradients.

5 Conclusion

We developed NVI, a framework that combines nested importance sampling and variational inference by optimizing a variational objective at every level of nesting. This formulation allows us to learn proposals and intermediate densities for a general class of samplers, which admit most commonly used importance sampling strategies as special cases. Our experiments demonstrate that samplers trained with NVI are able to outperform baselines when sampling from multimodal densities, Bayesian state-space models, and hierarchical deep generative models. Moreover, we found in our experiments that learning intermediate distributions results in better samplers.

References

Michael Arbel, Alexander GDG Matthews, and Arnaud Doucet. 2021. Annealed Flow Transport Monte Carlo. *arXiv preprint arXiv:2102.07501* (2021).

Jörg Bornschein and Yoshua Bengio. 2014. Reweighted wake-sleep. *arXiv preprint arXiv:1406.2751* (2014).

Jörg Bornschein and Yoshua Bengio. 2015. Reweighted Wake-Sleep. *International Conference on Learning Representations* (2015). arXiv:1406.2751

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. 2016. Importance Weighted Autoencoders. In *International Conference on Representations*. arXiv:1509.00519

N. Chopin. 2002. A Sequential Particle Filter Method for Static Models. *Biometrika* 89, 3 (Aug. 2002), 539–552. https://doi.org/10.1093/biomet/89.3.539

Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. 2006. Sequential Monte Carlo Samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68, 3 (June 2006), 411–436. https://doi.org/10.1111/j.1467-9868.2006.00553.x

Arnaud Doucet, Nando Freitas, and Neil Gordon (Eds.). 2001. *Sequential Monte Carlo Methods in Practice*. Springer New York, New York, NY. https://doi.org/10.1007/978-1-4757-3437-9

Matthew D Hoffman. 2017. Learning deep latent Gaussian models with Markov chain Monte Carlo. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 1510–1519.

Chin-Wei Huang, Shawn Tan, Alexandre Lacoste, and Aaron C Courville. 2018. Improving explorability in variational inference with annealed variational objectives. In *Advances in Neural Information Processing Systems*. 9701–9711.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

Dieterich Lawson, George Tucker, Christian A Naesseth, Chris Maddison, Ryan P Adams, and Yee Whye Teh. 2018. Twisted variational sequential monte carlo. In *Third workshop on Bayesian Deep Learning (NeurIPS)*.

Tuan Anh Le, Maximilian Igl, Tom Rainforth, Tom Jin, and Frank Wood. 2017. Auto-encoding sequential monte carlo. *arXiv preprint arXiv:1705.10306* (2017).

Tuan Anh Le, Maximilian Igl, Tom Rainforth, Tom Jin, and Frank Wood. 2018. Auto-Encoding Sequential Monte Carlo. In *International Conference on Learning Representations*. arXiv:1705.10306

Tuan Anh Le, A Kosiorek, N Siddharth, Yee Whye Teh, and Frank Wood. 2019. Revisiting reweighted wake-sleep for models with stochastic control flow. (2019).

Yingzhen Li, Richard E Turner, and Qiang Liu. 2017. Approximate inference with amortised mcmc. *arXiv preprint arXiv:1702.08343* (2017).

Gabriel Loaiza-Ganem and John P Cunningham. 2019. The continuous Bernoulli: fixing a pervasive error in variational autoencoders. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2019/file/f82798ec8909d23e55679ee26bb26437-Paper.pdf

Chris J Maddison, Dieterich Lawson, George Tucker, Nicolas Heess, Mohammad Norouzi, Andriy Mnih, Arnaud Doucet, and Yee Whye Teh. 2017. Filtering variational objectives. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 6576–6586.

Christian Naesseth, Scott Linderman, Rajesh Ranganath, and David Blei. 2018. Variational sequential monte carlo. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 968–977.

Christian Naesseth, Fredrik Lindsten, and Thomas Schon. 2015. Nested Sequential Monte Carlo Methods. In *International Conference on Machine Learning*. 1292–1301.

Christian A Naesseth, Scott W Linderman, Rajesh Ranganath, and David M Blei. 2017. Variational sequential monte carlo. *arXiv preprint arXiv:1705.11140* (2017).

Christian A. Naesseth, Fredrik Lindsten, and Thomas B. Schön. 2019. Elements of Sequential Monte Carlo. *arXiv:1903.04797 [cs, stat]* (March 2019). arXiv: 1903.04797.

Radford M Neal. 2001. Annealed importance sampling. *Statistics and computing* 11, 2 (2001), 125–139.

Tom Rainforth, Adam Kosiorek, Tuan Anh Le, Chris Maddison, Maximilian Igl, Frank Wood, and Yee Whye Teh. 2018. Tighter Variational Bounds Are Not Necessarily Better. In *International Conference on Machine Learning*. 4277–4285.

Rajesh Ranganath, Sean Gerrish, and David Blei. 2014. Black Box Variational Inference. In *Artificial Intelligence and Statistics*. 814–822.

Danilo Rezende and Shakir Mohamed. 2015. Variational Inference with Normalizing Flows. In *International Conference on Machine Learning*. 1530–1538.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 31st International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Eric P. Xing and Tony Jebara (Eds.), Vol. 32. PMLR, Bejing, China, 1278–1286.

Daniel Ritchie, Paul Horsfall, and Noah D. Goodman. 2016a. Deep Amortized Inference for Probabilistic Programs. *arXiv:1610.05735 [cs, stat]* (Oct. 2016). arXiv:cs, stat/1610.05735

Daniel Ritchie, Anna Thomas, Pat Hanrahan, and Noah D Goodman. 2016b. Neurally-guided procedural models: learning to guide procedural models with deep neural networks. *arXiv preprint arXiv: 1603.06143* (2016).

Geoffrey Roeder, Yuhuai Wu, and David K Duvenaud. 2017. Sticking the Landing: Simple, Lower-Variance Gradient Estimators for Variational Inference. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/e91068fff3d7fa1594dfdf3b4308433a-Paper.pdf

Tim Salimans, Diederik Kingma, and Max Welling. 2015. Markov chain monte carlo and variational inference: Bridging the gap. In *International Conference on Machine Learning*. 1218–1226.

John Schulman, Nicolas Heess, Theophane Weber, and Pieter Abbeel. 2015. Gradient Estimation Using Stochastic Computation Graphs. In *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (Eds.). Curran Associates, Inc., 3528–3536.

George Tucker, Dieterich Lawson, Shixiang Gu, and Chris J Maddison. 2018. Doubly reparameterized gradient estimators for monte carlo objectives. *arXiv preprint arXiv:1810.04152* (2018).

Tongzhou Wang, Yi Wu, Dave Moore, and Stuart J Russell. 2018. Meta-learning MCMC proposals. In *Advances in Neural Information Processing Systems*. 4146–4156.

David Wingate and Theo Weber. 2013. Automated Variational Inference in Probabilistic Programming. *arXiv preprint arXiv:1301.1299* (2013), 1–7. arXiv:1301.1299

Hao Wu, Heiko Zimmermann, Eli Sennesh, Tuan Anh Le, and Jan-Willem van de Meent. 2019. Amortized Population Gibbs Samplers with Neural Sufficient Statistics. *arXiv preprint arXiv:1911.01382* (2019).

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] - Appendix
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] - Appendix
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No] - Appendix
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]