
Reparameterized Variational Rejection Sampling

Martin Jankowiak
Generate Biomedicines

Du Phan
Google Research

Abstract

Traditional approaches to variational inference rely on parametric families of variational distributions, with the choice of family playing a critical role in determining the accuracy of the resulting posterior approximation. Simple mean-field families often lead to poor approximations, while rich families of distributions like normalizing flows can be difficult to optimize and usually do not incorporate the known structure of the target distribution due to their black-box nature. To expand the space of flexible variational families, we revisit Variational Rejection Sampling (VRS) (Grover et al., 2018), which combines a parametric proposal distribution with rejection sampling to define a rich non-parametric family of distributions that explicitly utilizes the known target distribution. By introducing a low-variance reparameterized gradient estimator for the parameters of the proposal distribution, we make VRS an attractive inference strategy for models with continuous latent variables. We argue theoretically and demonstrate empirically that the resulting method—Reparameterized Variational Rejection Sampling (RVRS)—offers an attractive trade-off between computational cost and inference fidelity. In experiments we show that our method performs well in practice and that it is well-suited for black-box inference, especially for models with local latent variables.

1 INTRODUCTION

Variational inference is a powerful method for approximate Bayesian inference with a number of appealing

properties, including support for data subsampling and model learning (Blei et al., 2017). Unfortunately, simple variational families like mean-field gaussian distributions often result in poor posterior approximations, while defining custom parametric families that better reflect the correlation structure and tail behavior of the exact posterior can be difficult, even for experts. This has motivated research into more flexible variational methods, including black-box methods like normalizing flows (Rezende and Mohamed, 2015) as well as hybrid methods that incorporate Markov Chain Monte Carlo (MCMC) (Salimans et al., 2015).

While these methods are powerful, they come with several disadvantages. Normalizing flows can be difficult to optimize, exhibit tail behavior that is difficult to control (Jaini et al., 2020), and introduce a large design space characterized by many hard-to-set hyperparameters. Moreover, due to their black-box nature normalizing flows typically do not incorporate the known structure of the target distribution. This is arguably a lost opportunity, especially in the context of probabilistic programming systems, where this information is readily available. The most powerful methods that combine variational inference with MCMC are gradient-based (Geffner and Domke, 2021; Zhang et al., 2021; Thin et al., 2021), with the result that many (possibly expensive) gradient steps may be required to generate a single sample. Moreover, good performance relies on carefully tuning the MCMC kernel, which can be challenging, since posterior curvature can vary considerably across latent space. In addition, these approaches typically introduce auxiliary latent variables, leading to a looser and more stochastic variational bound.

These considerations lead us to revisit a conceptually simpler hybrid variational inference method dubbed Variational Rejection Sampling (VRS) (Grover et al., 2018). Like MCMC-based methods, the target distribution is directly incorporated into the definition of the variational family, resulting in a non-parametric variational distribution. Since, however, rejection sampling is much simpler than MCMC, the result is a considerably simpler hybrid variational method that does not require delicate tuning or differentiating through long

Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238. Copyright 2024 by the author(s).

MCMC chains. Unfortunately, VRS utilizes score function (i.e. REINFORCE-like (Williams, 1992)) gradient estimators, which are known to be high variance, thus limiting its usefulness to discrete latent variable models, which are in any case not amenable to the reparameterization trick. In this work we set out to show that by introducing a reparameterized gradient estimator VRS becomes an attractive inference strategy for continuous latent variable models.

In summary our contributions include the following:

1. We introduce a reparameterized gradient estimator for VRS.
2. We show that the resulting method—RVRS—is especially well-suited for local latent variable models, including hierarchical models that additionally include global latent variables.
3. We characterize the variational gap of (R)VRS as a function of the rejection threshold parameter T .

2 PROBLEM SETTING

We are given a model with joint density of the form $p_{\theta}(\mathbf{x}, \mathbf{z}) = p_{\theta}(\mathbf{x}|\mathbf{z})p_{\theta}(\mathbf{z})$ where the latent variable $\mathbf{z} \in \mathbb{R}^D$ is governed by a prior $p_{\theta}(\mathbf{z})$ and \mathbf{x} in the likelihood $p_{\theta}(\mathbf{x}|\mathbf{z})$ represents observed data. We aim to devise a flexible variational approximation to the posterior $p_{\theta}(\mathbf{z}|\mathbf{x})$ that can be learned with a low-variance ELBO gradient estimator. Initially we do not assume any particular conditional independence structure, but in Sec. 4.4 we turn our attention to hierarchical models with both global and local latent variables, which benefit from additional consideration. We would like our method to be generic in nature so that it is suitable for black-box inference in a probabilistic programming framework. Additionally we would like our method to support model learning, i.e. learning θ in conjunction with the approximate posterior.

3 BACKGROUND

3.1 Variational inference

The most common variant of variational inference introduces a parametric variational distribution $q_{\phi}(\mathbf{z})$ and proceeds to optimize the parameters ϕ to minimize the Kullback-Leibler (KL) divergence between $q_{\phi}(\mathbf{z})$ and the posterior $p_{\theta}(\mathbf{z}|\mathbf{x})$, i.e. $\text{KL}(q_{\phi}(\mathbf{z})||p_{\theta}(\mathbf{z}|\mathbf{x}))$. This can be done by maximizing the Evidence Lower Bound or ELBO objective \mathcal{L} where

$$\mathcal{L} \equiv \mathbb{E}_{q_{\phi}(\mathbf{z})}[\log p_{\theta}(\mathbf{x}, \mathbf{z}) - \log q_{\phi}(\mathbf{z})] \leq \log p_{\theta}(\mathbf{x}) \quad (1)$$

and $\log p_{\theta}(\mathbf{x}) \equiv \log \mathbb{E}_{p_{\theta}(\mathbf{z})} [p_{\theta}(\mathbf{x}|\mathbf{z})]$. Thanks to the inequality in Eqn. 1 the ELBO naturally enables joint

model learning and inference, i.e. we can maximize the ELBO w.r.t. both variational parameters ϕ and model parameters θ simultaneously. As noted in the introduction, a potential shortcoming of this fully parametric approach is the difficulty of specifying suitable parameterizations for $q_{\phi}(\mathbf{z})$. For additional background see e.g. (Blei et al., 2017).

3.2 Variational Rejection Sampling

The basic idea behind VRS is simple: define a flexible variational distribution by taking a parametric proposal distribution $q_{\phi}(\mathbf{z})$ and warping it towards the posterior $p_{\theta}(\mathbf{z}|\mathbf{x})$ via a smoothed variant of rejection sampling. In more detail, define the variational distribution as

$$r_{\phi, \theta}(\mathbf{z}) \equiv \frac{q_{\phi}(\mathbf{z})a_{\phi, \theta}(\mathbf{z})}{\mathcal{Z}_r} \quad \text{with } \mathcal{Z}_r \equiv \int d\mathbf{z} q_{\phi}(\mathbf{z})a_{\phi, \theta}(\mathbf{z}) \quad (2)$$

where

$$\begin{aligned} a_{\phi, \theta}(\mathbf{z}) &\equiv \sigma(\log p_{\theta}(\mathbf{x}, \mathbf{z}) - \log q_{\phi}(\mathbf{z}) + T) \\ &= \sigma(-\ell_{\theta, \phi}^T(\mathbf{z})) \end{aligned} \quad (3)$$

is an acceptance probability with $a_{\phi, \theta}(\mathbf{z}) \in [0, 1]$ and $\sigma(\cdot)$ is the logistic function. We have also introduced the threshold parameter $T \in \mathbb{R}$ and the T -shifted log ratio $\ell_{\theta, \phi}^T(\mathbf{z})$. As $T \rightarrow \infty$ we have $a_{\phi, \theta}(\mathbf{z}) \rightarrow 1$ and $r_{\phi, \theta}(\mathbf{z}) \rightarrow q_{\phi}(\mathbf{z})$, recovering conventional variational inference with $q_{\phi}(\mathbf{z})$ as the variational distribution. In the opposite limit $T \rightarrow -\infty$ the acceptance probability is low, $a_{\phi, \theta}(\mathbf{z}) \rightarrow 0$, and $r_{\phi, \theta}(\mathbf{z}) \rightarrow p_{\theta}(\mathbf{z}|\mathbf{x})$. For intermediate T (i.e. T which leads to a few but not many rejected samples) we get a $r_{\phi, \theta}(\mathbf{z})$ that is closer to the posterior $p_{\theta}(\mathbf{z}|\mathbf{x})$ than the proposal distribution $q_{\phi}(\mathbf{z})$ at the cost of a moderate amount of additional computation. Indeed as shown in Grover et al. (2018), as T decreases for fixed $q_{\phi}(\mathbf{z})$ the ELBO increases monotonically and thus $\text{KL}(r_{\phi, \theta}(\mathbf{z})||p_{\theta}(\mathbf{z}|\mathbf{x}))$ decreases monotonically.

3.2.1 Sampling

Since $a_{\phi, \theta}(\mathbf{z}) \in [0, 1]$ it is straightforward to sample from $r_{\phi, \theta}(\mathbf{z})$, see Algorithm 1. The expected number of draws from the proposal distribution is given by \mathcal{Z}_r^{-1} , see Sec. C.1. For this reason we expect the sweet spot for VRS to occur for moderate values of $\mathcal{Z}_r^{-1} \sim 3 - 20$, where the cost of rejection sampling is not too high, but where the proposal distribution is still significantly ‘sculpted’ towards the posterior.

3.2.2 Gradient estimators

VRS is only practical if we can use gradient methods to optimize the corresponding ELBO given by

$$\mathcal{L}(\phi, \theta) = \mathbb{E}_{r_{\phi, \theta}(\mathbf{z})} [\log p_{\theta}(\mathbf{x}, \mathbf{z}) - \log r_{\phi, \theta}(\mathbf{z})] \quad (4)$$

Algorithm 1 Sampler for $r_{\phi, \theta}(\mathbf{z})$. **Input:** acceptance probability $a_{\phi, \theta}(\mathbf{z})$ and proposal $q_{\phi}(\mathbf{z})$.

```

1: while True do
2:    $\mathbf{z} \sim q_{\phi}(\mathbf{z})$ 
3:   if  $u < a_{\phi, \theta}(\mathbf{z})$  where  $u \sim \text{Uniform}(0, 1)$  then
4:     return  $\mathbf{z}$ 
5:   end if
6: end while
    
```

As shown in Grover et al. (2018), gradients for the parameters ϕ that define the proposal distribution $q_{\phi}(\mathbf{z})$ can be computed using the following estimator

$$\nabla_{\phi} \mathcal{L} = \text{COV}_{r_{\phi, \theta}(\mathbf{z})} [\mathcal{A}(\mathbf{z}), a_{\phi, \theta}(\mathbf{z}) \nabla_{\phi} \log q_{\phi}(\mathbf{z})] \quad (5)$$

with

$$\mathcal{A}(\mathbf{z}) \equiv \log p_{\theta}(\mathbf{x}, \mathbf{z}) - \log q_{\phi}(\mathbf{z}) - \log a_{\phi, \theta}(\mathbf{z}) \quad (6)$$

and where $\text{COV}_{r_{\phi, \theta}(\mathbf{z})}[A(\mathbf{z}), B(\mathbf{z})]$ denotes the covariance between random variables A and B w.r.t. the distribution $r_{\phi, \theta}(\mathbf{z})$. Similarly the gradient estimator for the model parameters θ is given by

$$\begin{aligned} \nabla_{\theta} \mathcal{L} = & \mathbb{E}_{r_{\phi, \theta}(\mathbf{z})} [\nabla_{\theta} \log p_{\theta}(\mathbf{x}, \mathbf{z})] - \\ & \text{COV}_{r_{\phi, \theta}(\mathbf{z})} [\mathcal{A}(\mathbf{z}), (1 - a_{\phi, \theta}(\mathbf{z})) \nabla_{\theta} \log p_{\theta}(\mathbf{x}, \mathbf{z})] \end{aligned} \quad (7)$$

It is easy to show (see Sec. C.2) that in the limit that $a_{\phi, \theta}(\mathbf{z}) \rightarrow 1$ and $r_{\phi, \theta}(\mathbf{z}) \rightarrow q_{\phi}(\mathbf{z})$ the gradient estimator Eqn. 5 reduces to a conventional score function (i.e. REINFORCE-like) gradient estimator, which is known to exhibit high variance, essentially due to its coarse credit assignment (Mohamed et al., 2020). It is straightforward to compute unbiased Monte Carlo estimates of Eqn. 5 and Eqn. 7, although doing so requires drawing $S > 1$ samples from $r_{\phi, \theta}(\mathbf{z})$ due to the covariance terms, see Sec. C.3.

4 REPARAMETERIZED VARIATIONAL REJECTION SAMPLING

The REINFORCE-like covariance term in Eqn. 5 is generally expected to be high variance and thus limit the applicability of VRS. Fortunately, as we show in Prop. 1, the VRS ELBO admits a reparameterized (i.e. pathwise) gradient estimator for ϕ if $q_{\phi}(\mathbf{z})$ is reparameterizable—a surprising capability, since $r_{\phi, \theta}(\mathbf{z})$ is not readily reparameterizable itself. Since the suite of reparameterizable proposal distributions is quite large—including e.g. Normal distributions, Dirichlet distributions, and normalizing flows with reparameterizable base distributions—the RVRS distribution $r_{\phi, \theta}(\mathbf{z})$ is quite flexible.

Proposition 1 *If the proposal distribution $q_{\phi}(\mathbf{z})$ is reparameterizable, then the VRS ELBO Eqn. 4 admits the following reparameterized gradient estimator for ϕ gradients*

$$\nabla_{\phi} \mathcal{L} = \mathbb{E}_{r_{\phi, \theta}(\mathbf{z})} \left[\left(2\bar{\mathcal{A}}(\mathbf{z}) \frac{\partial a_{\phi, \theta}(\mathbf{z})}{\partial \mathbf{z}} + a_{\phi, \theta}(\mathbf{z}) \frac{\partial \mathcal{A}(\mathbf{z})}{\partial \mathbf{z}} \right) \cdot \nabla_{\phi} \mathbf{z} \right] \quad (8)$$

where $\bar{\mathcal{A}}(\mathbf{z})$ is defined as $\bar{\mathcal{A}}(\mathbf{z}) \equiv \mathcal{A}(\mathbf{z}) - \mathbb{E}_{r_{\phi, \theta}(\mathbf{z}')} [\mathcal{A}(\mathbf{z}')]$ and $\nabla_{\phi} \mathbf{z}$ is the velocity field¹ corresponding to infinitesimal displacement of $q_{\phi}(\mathbf{z})$ in ϕ -space. Eqn. 8 reduces to a conventional reparameterized gradient in the limit that $a_{\phi, \theta}(\mathbf{z}) \rightarrow 1$ and $r_{\phi, \theta}(\mathbf{z}) \rightarrow q_{\phi}(\mathbf{z})$. See Sec. A for the proof and additional details.²

Fundamentally the existence of a pathwise gradient estimator can be traced to three properties of $r_{\phi, \theta}(\mathbf{z})$: i) $r_{\phi, \theta}(\mathbf{z})$ is proportional to a reparameterizable distribution, namely $q_{\phi}(\mathbf{z})$; ii) $r_{\phi, \theta}(\mathbf{z})$ depends on ϕ only through $q_{\phi}(\mathbf{z})$; and iii) we can compute $a_{\phi, \theta}(\mathbf{z})$ and its gradients pointwise. We note that the derivation of Prop. 1 is conceptually similar to that behind ‘doubly reparameterized gradients’ (Tucker et al., 2018), although in that case a gradient estimator that is *al-ready* reparameterized is manipulated to transform a score-function-like term to further reduce variance.

4.1 Model parameter gradients

Unfortunately it seems unlikely that the covariance term in Eqn. 7 can be reparameterized in a straightforward way, since eliminating $\nabla_{\theta} \log p_{\theta}(\mathbf{x}, \mathbf{z})$ would require e.g. a reparameterized sampler of $p_{\theta}(\mathbf{z}|\mathbf{x})$. However, we show empirically that this term can be safely dropped at the cost of introducing some bias.³ This is because this term encodes how the log evidence estimate changes due to changes in $\ell_{\theta, \phi}^T(\mathbf{z})$ in Eqn. 3—i.e. how model parameters modulate the acceptance probability and thus the resulting rejection-sampled variational distribution—and not the ‘direct’ change encoded by the term $\mathbb{E}_{r_{\phi, \theta}(\mathbf{z})} [\nabla_{\theta} \log p_{\theta}(\mathbf{x}, \mathbf{z})]$.

4.2 Adapting the threshold T

Choosing an appropriate value of T in the vicinity of $\mathbb{E}_{q_{\phi}(\mathbf{z})} [\log q_{\phi}(\mathbf{z}) - \log p_{\theta}(\mathbf{x}, \mathbf{z})]$ is crucial for good performance of (R)VRS. In Grover et al. (2018) the authors propose a strategy based on quantiles of

¹For example if $q_{\phi}(z) = \mathcal{N}(z|\mu, \sigma^2)$ then $\nabla_{\mu} z = 1$ and $\nabla_{\sigma} z = (z - \mu)/\sigma$.

²In particular in Sec. A.2 we describe how we leverage automatic differentiation and $S > 1$ samples from $r_{\phi, \theta}(\mathbf{z})$ to obtain an unbiased Monte Carlo estimate of Eqn. 8.

³This is a classic bias/variance trade-off: this term can be retained (or re-introduced towards the end of optimization) if the bias is a concern.

$\log p_{\theta}(\mathbf{x}, \mathbf{z})/q_{\phi}(\mathbf{z})$. While we find that this strategy can work, we prefer a gradient-based strategy for tuning the threshold parameter T that allows direct control over the computational cost of (R)VRS. Another advantage of this approach is that because it is gradient-based it offers the possibility of choosing T using amortized inference, although we do not explore that possibility here. Recall that $\mathcal{Z}_r \equiv \int d\mathbf{z} q_{\phi}(\mathbf{z}) a_{\phi, \theta}(\mathbf{z})$ is the mean acceptance probability of the rejection sampler and consider the loss $\mathcal{L}(T) = \frac{1}{2} (\mathcal{Z}_r - \mathcal{Z}_{\text{tgt}})^2$ where $\mathcal{Z}_{\text{tgt}} \in (0, 1)$ is a target acceptance probability. Then the gradient $\frac{\partial \mathcal{L}}{\partial T}$ is given by

$$\frac{\partial \mathcal{L}}{\partial T} = \mathbb{E}_{q_{\phi}(\mathbf{z})} [a_{\phi, \theta}(\mathbf{z}) - \mathcal{Z}_{\text{tgt}}] \mathbb{E}_{q_{\phi}(\mathbf{z})} [a_{\phi, \theta}(\mathbf{z})(1 - a_{\phi, \theta}(\mathbf{z}))] \quad (9)$$

which we can readily compute unbiased estimates of, since we have $S > 1$ samples at our disposal. Throughout this work we use Monte Carlo estimates of $\frac{\partial \mathcal{L}}{\partial T}$ to tune T ; see Sec. D in the supplement for details.

4.3 Models with only local latent variables

For models with only local latent variables like a VAE (Kingma and Welling, 2013) sampling, ELBO estimation, and ELBO gradient estimation for RVRS trivially factorize across data points, and thus RVRS admits unbiased mini-batch learning for such models. An efficient sampler for RVRS in this scenario requires a flexible rejection sampling scheme that maximizes usage of computational resources. In particular during training we can choose between: i) an unbiased sampler that terminates when $S > 1$ latent samples have been generated for every data point; and ii) a (potentially much) faster biased sampler that terminates after generating a *fixed* number of proposals for each data point. See Algorithm 2 & Algorithm 3 in the supplement for details. As we report in Fig. 6 in Sec. G the small bias introduced by the faster sampler has a correspondingly small impact on performance.

4.4 Hierarchical models with global and local latent variables

We now consider models with both a global latent variable \mathbf{z}_G and local latent variables $\{\mathbf{z}_n\}$, with $n = 1, \dots, N$ indexing the N observed data points $\{\mathbf{x}_n\}$. We assume the following conditional independence structure:

$$p_{\theta}(\mathbf{x}_{1:N}, \mathbf{z}_G, \mathbf{z}_{1:N}) = p_{\theta}(\mathbf{z}_G) \prod_{n=1}^N p_{\theta}(\mathbf{x}_n | \mathbf{z}_n, \mathbf{z}_G) p(\mathbf{z}_n | \mathbf{z}_G)$$

While RVRS can be applied to the joint latent space $\{\mathbf{z}_G, \mathbf{z}_{1:N}\}$, the resulting algorithm does not admit unbiased data subsampling (i.e. mini-batch learning), since

$a_{\phi, \theta}(\mathbf{z})$ depends on the entire dataset, limiting this approach to moderate N .⁴ To enable data subsampling we adopt a hybrid approach in which the posterior over \mathbf{z}_G is approximated by a parametric distribution $q_{\phi}(\mathbf{z}_G)$ while the conditional posteriors $p_{\theta}(\mathbf{z}_n | \mathbf{z}_G, \mathbf{x}_n)$ are approximated by RVRS. This can be understood as an instance of a ‘locally enhanced bound’ (Geffner and Domke, 2022), and is analogous to the ‘Semi-DAIS’ approach explored in Jankowiak and Phan (2022) in the context of UHA/DAIS. We refer to this semi-parametric approach as Semi-RVRS. See Sec. E for details.

5 CONVERGENCE ANALYSIS

It is evident from the structure of $r_{\phi, \theta}(\mathbf{z})$ in Eqn. 2 that as $T \rightarrow -\infty$ the variational distribution $r_{\phi, \theta}(\mathbf{z})$ converges to the exact posterior $p_{\theta}(\mathbf{z} | \mathbf{x})$ *pointwise*. But can we say anything about the corresponding ELBO in Eqn. 4? As we would expect, the variational gap goes to zero in the same limit as e^T , see Prop. 2. Notably the relative simplicity of rejection sampling allows us to prove a generic result, whereas an analogous result for DAIS in (Zhang et al., 2021) is limited to linear Gaussian models due to the complexity of analyzing MCMC chains.

Proposition 2 (A) *Assume that $q_{\phi}(\mathbf{z})$ is sufficiently heavy-tailed so that $\xi \equiv \mathbb{E}_{p_{\theta}(\mathbf{z} | \mathbf{x})} \left[\frac{p_{\theta}(\mathbf{x}, \mathbf{z})}{q_{\phi}(\mathbf{z})} \right]$ is finite. Then the variational gap Δ between $\log p_{\theta}(\mathbf{x})$ and the ELBO is bounded from above as $\Delta < \frac{3}{2} e^T \xi$ for $T < -\log 2\xi$. (B) An analogous bound holds for the hierarchical modeling case considered in Sec. 4.4, where the bound includes an additional term $\text{KL}(q_{\phi}(\mathbf{z}_G) || p_{\theta}(\mathbf{z}_G | \mathbf{x}_{1:N}))$ that encodes the suboptimality of the parametric variational approximation for the global latent variable \mathbf{z}_G . For additional details and the proof see Sec. B in the supplement.*

6 RELATED WORK

Many variational objectives that go beyond a conventional ELBO have been proposed in the literature. These include the importance weighted autoencoder (IWAE) (Burda et al., 2015; Cremer et al., 2017), the thermodynamic variational objective (Masrani et al., 2019), and approaches that make use of Sequential Monte Carlo (Le et al., 2017; Maddison et al., 2017; Naesseth et al., 2018). Variational Rejection Sampling (VRS) was proposed by Grover et al. (2018) and applied to models with discrete latent variables. An early

⁴This is of course equally true of other non-parametric approaches like UHA/DAIS (Geffner and Domke, 2021; Zhang et al., 2021), although see (Jankowiak and Phan, 2022).

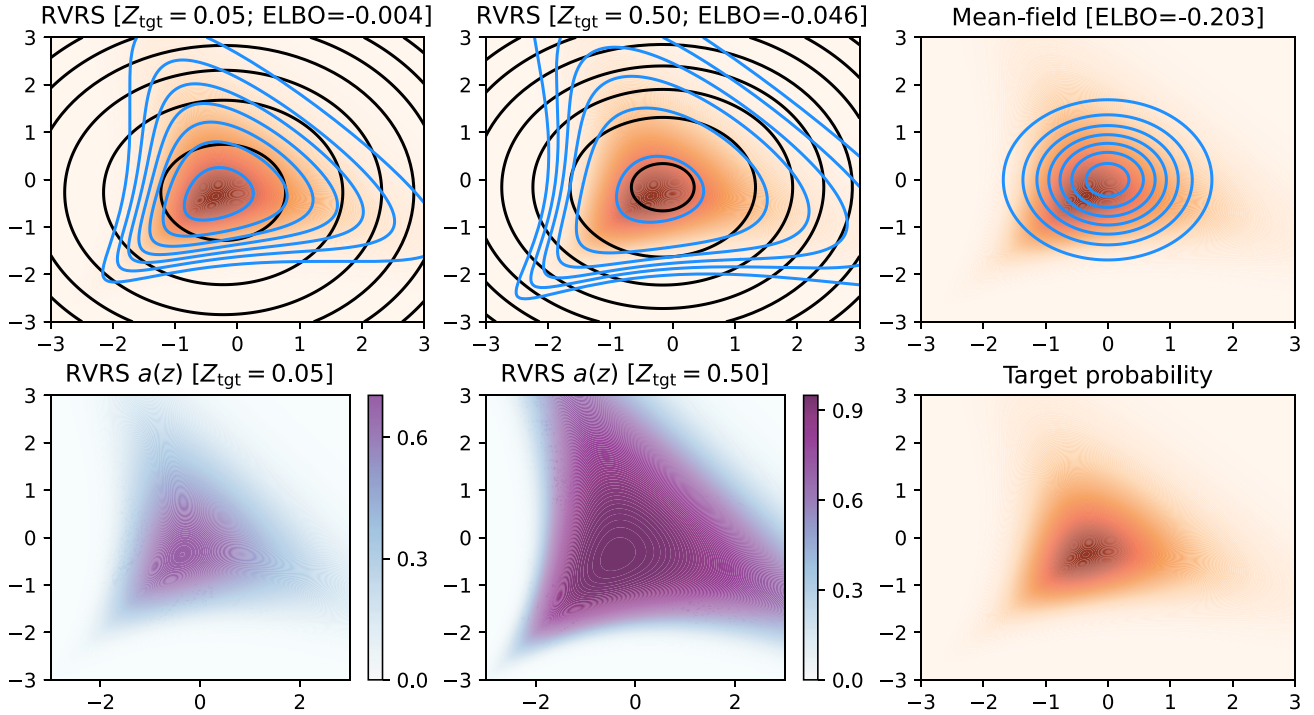


Figure 1: We illustrate how RVRS works on a (normalized) funnel-shaped target distribution (orange density). Blue contours depict variational fits, with a mean-field Normal fit depicted in the upper right figure. The first two columns depict RVRS fits for $Z_{\text{tgt}} \in \{0.05, 0.5\}$, with black contours depicting mean-field Normal proposal distributions $q_{\phi}(\mathbf{z})$. The leftmost figures in the lower row depict the acceptance probability $a_{\phi, \theta}(\mathbf{z})$; for $Z_{\text{tgt}} = 0.05$ $a_{\phi, \theta}(\mathbf{z})$ differs significantly from 1 everywhere so that $r_{\phi, \theta}(\mathbf{z})$ is strongly sculpted towards the target and the ELBO is nearly optimal (i.e. close to 0).

combination of MCMC methods with variational inference was proposed by Salimans et al. (2015) and Wolf et al. (2016) and has led to follow-up work by many authors (Hoffman, 2017; Caterini et al., 2018; Ruiz and Titsias, 2019). Arguably the most powerful hybrid variational methods proposed so far are those that incorporate gradient-based MCMC like Uncorrected Hamiltonian Annealing (UHA; (Geffner and Domke, 2021)) and the essentially identical algorithm Differentiable Annealed Importance Sampling (DAIS; (Zhang et al., 2021)); see also (Thin et al., 2021; Doucet et al., 2022; Matthews et al., 2022). A conceptually related but distinct gradient-based approach utilizes ergodic maps built with Hamiltonian dynamics to formulate flexible variational distributions (Xu et al., 2023; Xu and Campbell, 2023). For a recent review of some of these methods see Doucet et al. (2023). Another important line of work has seen the development of rich parametric families of distributions like normalizing flows for use in variational inference (Rezende and Mohamed, 2015; Kingma et al., 2016; Papamakarios et al., 2021). Rejection sampling has seen other applications in probabilistic machine learning. For example Stimper

et al. (2022) adapt earlier work (Bauer and Mnih, 2019) to build normalizing flows where the base distribution is defined via a learned rejection sampling scheme. Indeed Stimper et al. (2022) use a REINFORCE-like gradient estimator modified from VRS that could benefit from our reparameterized estimator in Prop. 1. The truncated rejection sampling scheme that underlies these works has inspired other applications, including e.g. the ‘energy-inspired’ models in Lawson et al. (2019). In addition other researchers have shown how to formulate reparameterized gradient estimators for sampling algorithms that are not obviously reparameterizable; for example Zoltowski et al. (2021) consider the case of slice sampling. Finally Naesseth et al. (2017) show how to construct partially reparameterized gradient estimators for distributions defined by classical rejection samplers (i.e. not the ‘smoothed’ variant in Eqn. 2).

7 EXPERIMENTS

All our experiments are implemented using JAX and NumPyro (Bradbury et al., 2020; Phan et al., 2019;

Bingham et al., 2019).⁵ We explore a number of different aspects of RVRS, including support for $D \gg 1$ latent dimensions and model learning (Sec. 7.3), variational auto-encoders (Sec. 7.4), and hierarchical models (Sec. 7.5). We provide additional experimental details and report additional results in Sec. F-G.

7.1 Characterizing RVRS

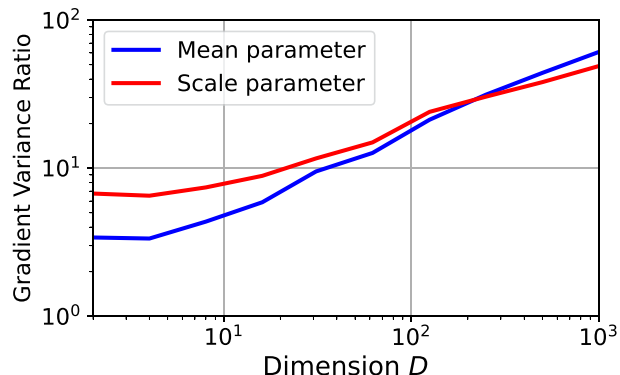


Figure 2: We compare RVRS and VRS gradient variance for a logistic regression problem with $N = 100$ data points as we vary the latent dimension D . The proposal $q_\phi(\mathbf{z})$ is mean-field Normal and we depict the ratio of gradient variances between VRS and RVRS for the mean and scale (i.e. square root variance) parameters of $q_\phi(\mathbf{z})$. See Sec. 7.1 for discussion.

We begin with a few experiments to characterize some of the general characteristics of RVRS. In Fig. 1 we illustrate graphically how RVRS sculpts a mean-field gaussian proposal distribution $q_\phi(\mathbf{z})$ to match a non-gaussian target. Notably a nearly optimal ELBO is achieved for $\mathcal{Z}_{\text{tgt}} = 0.05$.

Next we compare the variance of RVRS and VRS ELBO gradient estimators on a logistic regression model, see Fig. 2. We find that VRS gradient variance is always larger than in the case of RVRS—e.g. by a factor of ~ 15 for $D = 62$ latent dimensions—and that the ratio increases as the dimension increases. For an example of how large gradient variance negatively impacts the optimization performance of VRS see Fig. 8 in Sec. G.

Finally in Fig. 3 we explore how RVRS depends on the hyperparameter $\mathcal{Z}_{\text{tgt}} \in (0, 1)$. As we would expect the ELBO increases monotonically as \mathcal{Z}_{tgt} decreases—as it must if T adaptation and $q_\phi(\mathbf{z})$ learning are working correctly. Tellingly, we see that the width of the

⁵We plan to make an open-source implementation of RVRS available as a NumPyro AutoGuide at <https://num.pyro.ai>.

proposal distribution $q_\phi(\mathbf{z})$ increases as \mathcal{Z}_{tgt} decreases. This illustrates the basic principle that (R)VRS exploits to achieve better variational approximations. Since the ELBO tends to prefer variational distributions that excessively avoid low-density regions of the posterior, a common failure mode of parametric variational inference is to underestimate posterior uncertainty. Target-dependent rejection sampling offers a simple but effective mechanism to better capture posterior uncertainty: inflate the width of the proposal distribution where needed and reject a portion of samples in regions where the density of the proposal $q_\phi(\mathbf{z})$ is excessive (due to e.g. the parametric misfit of the proposal). The upshot is that (R)VRS can better capture both correlations and tail behavior and thus yield higher fidelity variational approximations.

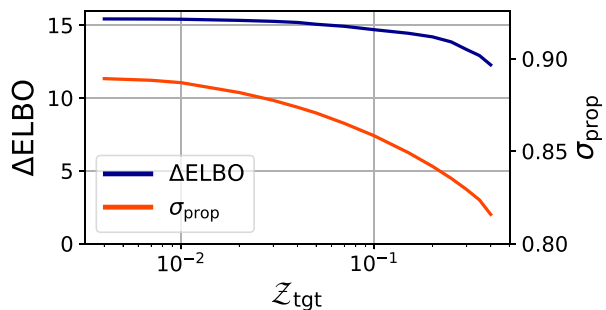


Figure 3: We explore the performance of RVRS as a function of \mathcal{Z}_{tgt} on a logistic regression problem in $D = 51$ dimensions. The blue curve depicts the ELBO improvement over a mean-field baseline, while the orange curve depicts the geometric mean of the D scales (i.e. square root variances) that define the mean-field Normal proposal $q_\phi(\mathbf{z})$. As $\mathcal{Z}_{\text{tgt}} \rightarrow 0$ the proposal distribution becomes broader, especially compared to the mean-field fit obtained with a standard ELBO, which yields $\sigma = 0.70$. See Sec. 7.1 for discussion.

7.2 Logistic regression

We compare RVRS to a large number of variational baselines on 5 logistic regression tasks. To ensure that posterior distributions are relatively non-gaussian we consider $N = 100$ data points, while the latent dimension ranges from $D = 15$ to $D = 58$. We consider three fully parametric baselines: mean-field with a factorized Normal distribution (**MF**); a multivariate Normal distribution (**MVN**); and a Block Neural Autoregressive normalizing flow (**Flow**; De Cao et al. (2020)). We also consider four non-parametric baselines: IWAE with $K \in \{8, 24\}$ particles (**IWAE**); and UHA with $K \in \{8, 24\}$ gradient steps (**UHA**). For RVRS we con-

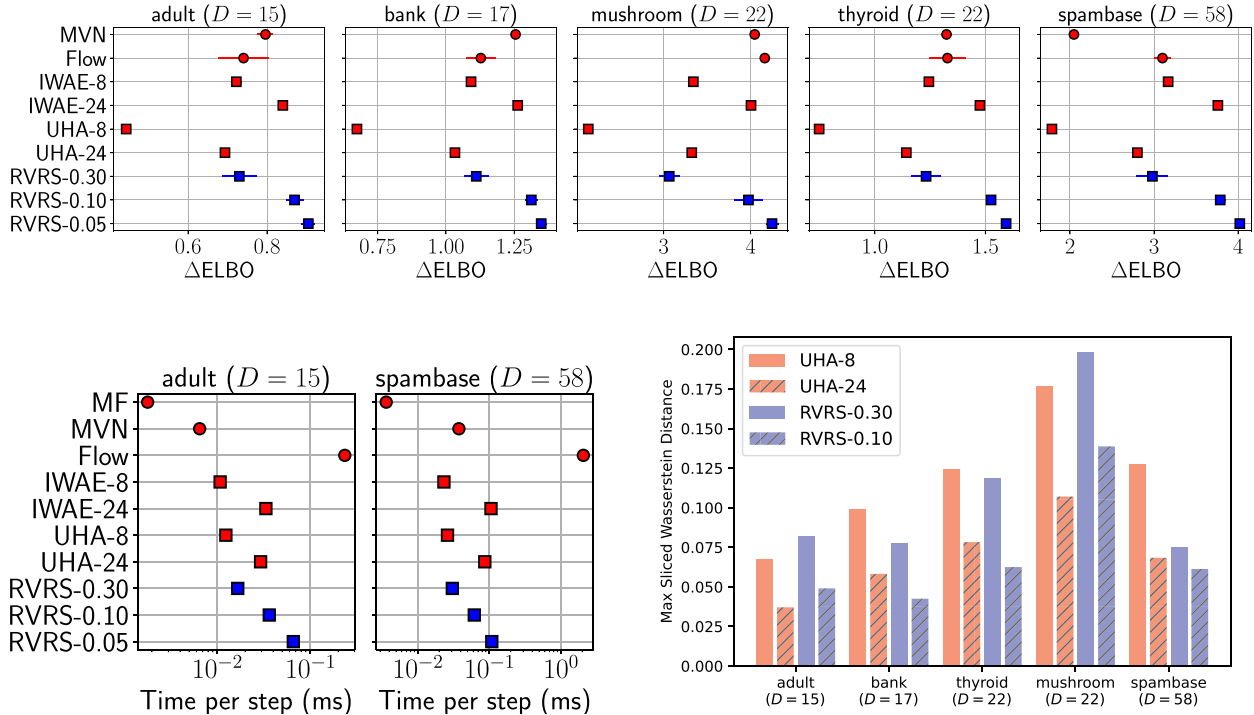


Figure 4: **(Top)** We depict ELBO improvements above the mean-field baseline for 9 variational methods on 5 logistic regression tasks. Circles and squares indicate parametric and non-parametric methods, respectively. Error bars denote two standard deviations and ELBOs are averaged across 5 runs. **(Bottom left)** We depict the corresponding gradient step times for two logistic regression tasks. **(Bottom right)** We compare the fidelity of posterior samples generated by UHA and RVRS w.r.t. the Max Sliced Wasserstein distance, using samples from NUTS as a reference. Here and elsewhere RVRS-0.50 refers to RVRS with $\mathcal{Z}_{\text{tgt}} = 0.5$, IWAE-8 refers to IWAE with $K = 8$ particles, UHA-8 refers to UHA with $K = 8$ gradient steps, etc. See Sec. 7.2 for discussion.

sider $\mathcal{Z}_{\text{tgt}} \in \{0.3, 0.1, 0.05\}$. For the results see Fig. 4.

We find that RVRS performs well across the board. For example RVRS with $\mathcal{Z}_{\text{tgt}} = 0.10$ outperforms the normalizing flow on 4/5 datasets but is much faster to train. Moreover RVRS-0.10 matches or exceeds the performance of IWAE with $K = 24$ particles on all datasets. The RVRS-0.10 ELBO also exceeds that of UHA with $K = 24$ steps on all datasets, but we note that this gap is probably at least partially explained by the additional looseness of the UHA variational bound. Indeed if we compare RVRS and UHA posterior samples to ‘gold standard’ samples obtained with NUTS (Hoffman et al., 2014; Carpenter et al., 2017) and use a Max Sliced Wasserstein distance (Deshpande et al., 2019) to quantify the fidelity of the posterior approximation, we find that RVRS-0.10 (respectively, RVRS-0.30) approximately matches the performance of UHA-24 (respectively, UHA-8), see Fig. 4.

7.3 Gaussian process classification

To probe the ability of RVRS to handle both model learning and higher-dimensional latent spaces, we consider Gaussian process models for binary classification. For each of 3 datasets we consider $N = 256$ data points and thus $D = 256$ latent dimensions. Model parameter θ are the $D_{\mathbf{x}} + 1$ kernel hyperparameters, where $D_{\mathbf{x}}$ is the dimension of the inputs with $D_{\mathbf{x}} \in \{18, 28, 51\}$. See Fig. 5 for the results. Perhaps surprisingly given the large dimension, we find that RVRS matches or exceeds the performance of the other methods. This is even true for IWAE with $K = 128$ particles. Notably UHA-6 does about the same as UHA-3, emphasizing the difficulty of optimizing the UHA ELBO—which effectively differentiates through a short MCMC chain—with its potential for numerical instability w.r.t. the step size and mass matrix that define the Hamiltonian dynamics. Thus although (high-dimensional) gradients offer a lot of information about the posterior density, effectively utilizing that information can be challenging to the point where rejection sampling—which we would

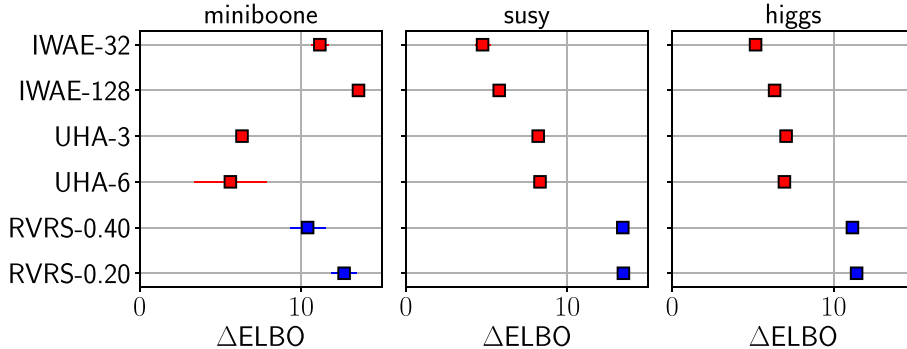


Figure 5: We depict ELBO improvements above the multivariate Normal baseline for 6 variational methods on 3 Gaussian process classification tasks. Results are averaged across five runs and error bars denote two standard deviations. See Sec. 7.3 for discussion.

generally expect to be less effective for large D —can be just as effective or even more so.

7.4 Variational autoencoders

We compare conventional ELBO training with RVRS, IWAE, and UHA/DAIS on a VAE (Kingma and Welling, 2013) trained on statically binarized MNIST. We use the same encoder-decoder architecture as in (Burda et al., 2015) and set the dimension of the latent variable to $D = 50$. See Table 1 for the results. Notably RVRS is faster than IWAE because RVRS only requires computing gradients through a small number ($S = 2$) of accepted samples. We find that RVRS consistently outperforms IWAE but is edged out by UHA with many gradient steps. The good performance of UHA with $K = 20$ gradient steps comes at significant computational cost, however, as training is $\sim 5.6x$ slower than RVRS-0.025. Thus an attractive feature of RVRS trained with Algorithm 3 is that it can make effective use of parallel hardware, while UHA is bottlenecked by the serial nature of MCMC chains.

7.5 Hierarchical modeling

We evaluate RVRS on a hierarchical model with local latent variables that was also considered by Jankowiak and Phan (2022). In detail we consider a Bayesian linear regressor that utilizes a Student’s t likelihood. Since this likelihood can be represented as a continuous mixture of Normal distributions, this choice corresponds to a hierarchical model with local Gamma latent variables that can be integrated out exactly. We compare three variational approaches, all of which use a mean-field Normal distribution for the global latent coefficient. We consider two semi-parametric approaches—Semi-DAIS (Jankowiak and Phan, 2022) and Semi-RVRS—that only differ in how the approximate posterior over the local latent variables is constructed. We also compare

against an oracle baseline obtained by integrating out the Gamma variates before performing variational inference. This oracle represents an upper performance bound on the two semi-parametric approaches. See Table 2 for results.⁶

For both datasets we find that Semi-RVRS nearly matches the performance of the oracle, even with $\mathcal{Z}_{\text{tgt}} = 0.5$, implying that Semi-RVRS yields a conditional posterior over the local latent variables that is nearly exact. Semi-DAIS also yields good performance but fails to approach the oracle upper bound even with $K = 32$ gradient steps. Thus this experiment highlights a particular strength of RVRS, namely dealing with low-dimensional latent variables, a regime in which methods based on expensive gradient-based MCMC can be overkill. Perhaps remarkably RVRS can still be competitive in higher dimensions, as demonstrated in the Gaussian process experiment in Sec. 7.3.

8 DISCUSSION

Given its relative simplicity, it is remarkable that RVRS can match—and in some cases exceed—the performance of more complex gradient-based hybrid variational inference schemes like UHA/DAIS (Geffner and Domke, 2021; Zhang et al., 2021). For this reason we believe it could be especially valuable to combine RVRS with other methods, since RVRS with moderate \mathcal{Z}_{tgt} provides a relatively cheap and simple way to achieve non-trivial refinement of the proposal distribution $q_{\phi}(\mathbf{z})$. For example it would be natural to use a normalizing flow or MixFlow (Xu et al., 2023) in place of a simple parametric proposal. This could

⁶Note that the Semi-RVRS results in Table 2 were obtained using the unbiased sampler Algorithm 2 during training. In Fig. 6 in Sec. G we provide a comparison to results obtained with the faster biased sampler defined in Algorithm 3, which yields nearly identical performance.

Method	Standard VAE	IWAE-10	IWAE-20	IWAE-40	UHA-10	UHA-20	RVRS-0.1	RVRS-0.05	RVRS-0.025
-ELBO	95.30 ± 0.14	91.21 ± 0.07	90.44 ± 0.07	89.81 ± 0.09	89.75 ± 0.09	88.46 ± 0.22	90.74 ± 0.16	90.00 ± 0.12	89.55 ± 0.12
ms / grad	0.70	1.06	1.49	1.97	5.17	9.73	1.19	1.36	1.75

Table 1: We report negative ELBO objectives (lower is better; mean ± standard deviation over 5 replicates) computed on held-out test data together with gradient step times for the VAE experiment in Sec. 7.4. In all cases we report results using the same objective used during training. Results are obtained with a RTX 2070 GPU.

Dataset	Semi-DAIS-8	Semi-DAIS-16	Semi-DAIS-32	Semi-RVRS-0.50	Semi-RVRS-0.10	Oracle
Pol	37.8 ± 1.6	73.8 ± 1.6	99.8 ± 1.6	116.9 ± 0.9	140.2 ± 0.9	143.1 ± 0.4
Bike	49.5 ± 0.8	110.8 ± 0.7	155.8 ± 0.8	187.8 ± 0.7	223.5 ± 0.7	228.6 ± 0.5

Table 2: We report ELBO improvement above a mean-field baseline for the hierarchical model in Sec. 7.5 (mean ± standard deviation). Results are averaged across 5 replicates.

be especially attractive in cases where there are diminishing returns to e.g. using more layers (in the case of normalizing flows) or more pushforwards (in the case of MixFlows). Importantly in RVRS we only need to differentiate through *accepted* samples $\mathbf{z} \sim r_{\phi, \theta}$, which limits the computational cost of leveraging RVRS. More broadly the design space of hybrid variational algorithms remains only partially explored and involves various algorithmic and computational trade-offs. As such we expect that RVRS could be a useful component in the design of future hybrid variational inference methods.

Acknowledgements

We warmly thank Matthew D. Hoffman for feedback on a draft manuscript. MJ’s contributions to the work reported here are independent of his role at Generate Biomedicines.

References

Arthur Asuncion and David Newman. Uci machine learning repository, 2007.

Matthias Bauer and Andriy Mnih. Resampled priors for variational autoencoders. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 66–75. PMLR, 2019.

Eli Bingham, Jonathan P Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D Goodman. Pyro: Deep universal probabilistic programming. *The Journal of Machine Learning Research*, 20(1):973–978, 2019.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal

Maclaurin, and Skye Wanderman-Milne. Jax: composable transformations of python+ numpy programs, 2018. URL <http://github.com/google/jax>, 4:16, 2020.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

Bob Carpenter, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus A Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. Stan: A probabilistic programming language. *Journal of statistical software*, 76, 2017.

Anthony L. Caterini, Arnaud Doucet, and Dino Sejdinovic. Hamiltonian variational auto-encoder. *CoRR*, abs/1805.11328, 2018. URL <http://arxiv.org/abs/1805.11328>.

Chris Cremer, Quaid Morris, and David Duvenaud. Reinterpreting importance-weighted autoencoders. *arXiv preprint arXiv:1704.02916*, 2017.

Nicola De Cao, Wilker Aziz, and Ivan Titov. Block neural autoregressive flow. In *Uncertainty in artificial intelligence*, pages 1263–1273. PMLR, 2020.

Ishan Deshpande, Yuan-Ting Hu, Ruoyu Sun, Ayis Pyrros, Nasir Siddiqui, Sanmi Koyejo, Zhizhen Zhao, David Forsyth, and Alexander G Schwing. Max-sliced wasserstein distance and its use for gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10648–10656, 2019.

Arnaud Doucet, Will Grathwohl, Alexander G Matthews, and Heiko Strathmann. Score-based diffusion meets annealed importance sampling. *Advances in Neural Information Processing Systems*, 35:21482–21494, 2022.

Arnaud Doucet, Eric Moulines, and Achille Thin. Differentiable samplers for deep latent variable models. *Philosophical Transactions of the Royal Society A*, 381(2247):20220147, 2023.

- Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z. Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, Léo Gautheron, Nathalie T.H. Gayraud, Hicham Janati, Alain Rakotomamonjy, Ievgen Redko, Antoine Rolet, Antony Schutz, Vivien Seguy, Danica J. Sutherland, Romain Tavenard, Alexander Tong, and Titouan Vayer. Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021. URL <http://jmlr.org/papers/v22/20-451.html>.
- Tomas Geffner and Justin Domke. Mcmc variational inference via uncorrected hamiltonian annealing. *Advances in Neural Information Processing Systems*, 34, 2021.
- Tomas Geffner and Justin Domke. Variational inference with locally enhanced bounds for hierarchical models. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 7310–7323. PMLR, 17–23 Jul 2022.
- Aditya Grover, Ramki Gummadi, Miguel Lazaro-Gredilla, Dale Schuurmans, and Stefano Ermon. Variational rejection sampling. In *International Conference on Artificial Intelligence and Statistics*, pages 823–832. PMLR, 2018.
- Matthew D Hoffman. Learning deep latent gaussian models with markov chain monte carlo. In *International conference on machine learning*, pages 1510–1519. PMLR, 2017.
- Matthew D Hoffman, Andrew Gelman, et al. The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *J. Mach. Learn. Res.*, 15(1):1593–1623, 2014.
- Priyank Jaini, Ivan Kobyzev, Yaoliang Yu, and Marcus Brubaker. Tails of lipschitz triangular flows. In *International Conference on Machine Learning*, pages 4673–4681. PMLR, 2020.
- Martin Jankowiak and Fritz Obermeyer. Pathwise derivatives beyond the reparameterization trick. In *International conference on machine learning*, pages 2235–2244. PMLR, 2018.
- Martin Jankowiak and Du Phan. Surrogate likelihoods for variational annealed importance sampling. In *International Conference on Machine Learning*, pages 9881–9901. PMLR, 2022.
- Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29, 2016.
- John Lawson, George Tucker, Bo Dai, and Rajesh Ranganath. Energy-inspired models: Learning with sampler-induced distributions. *Advances in Neural Information Processing Systems*, 32, 2019.
- Tuan Anh Le, Maximilian Igl, Tom Rainforth, Tom Jin, and Frank Wood. Auto-encoding sequential monte carlo. *arXiv preprint arXiv:1705.10306*, 2017.
- Chris J Maddison, John Lawson, George Tucker, Nicolas Heess, Mohammad Norouzi, Andriy Mnih, Arnaud Doucet, and Yee Teh. Filtering variational objectives. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
- Vaden Masrani, Tuan Anh Le, and Frank Wood. The thermodynamic variational objective. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.
- Alex Matthews, Michael Arbel, Danilo Jimenez Rezende, and Arnaud Doucet. Continual repeated annealed flow transport Monte Carlo. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 15196–15219. PMLR, 17–23 Jul 2022. URL <https://proceedings.mlr.press/v162/matthews22a.html>.
- Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte carlo gradient estimation in machine learning. *J. Mach. Learn. Res.*, 21(132):1–62, 2020.
- Christian Naesseth, Francisco Ruiz, Scott Linderman, and David Blei. Reparameterization gradients through acceptance-rejection sampling algorithms. In *Artificial Intelligence and Statistics*, pages 489–498. PMLR, 2017.
- Christian Naesseth, Scott Linderman, Rajesh Ranganath, and David Blei. Variational sequential monte carlo. In *International conference on artificial intelligence and statistics*, pages 968–977. PMLR, 2018.

- George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *The Journal of Machine Learning Research*, 22(1):2617–2680, 2021.
- Du Phan, Neeraj Pradhan, and Martin Jankowiak. Composable effects for flexible and accelerated probabilistic programming in numpyro. *arXiv preprint arXiv:1912.11554*, 2019.
- Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *Artificial intelligence and statistics*, pages 814–822. PMLR, 2014.
- Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.
- Francisco Ruiz and Michalis Titsias. A contrastive divergence for combining variational inference and mcmc. In *International Conference on Machine Learning*, pages 5537–5545. PMLR, 2019.
- Tim Salimans, Diederik Kingma, and Max Welling. Markov chain monte carlo and variational inference: Bridging the gap. In *International Conference on Machine Learning*, pages 1218–1226. PMLR, 2015.
- Vincent Stimper, Bernhard Schölkopf, and José Miguel Hernández-Lobato. Resampling base distributions of normalizing flows. In *International Conference on Artificial Intelligence and Statistics*, pages 4915–4936. PMLR, 2022.
- Achille Thin, Nikita Kotelevskii, Arnaud Doucet, Alain Durmus, Eric Moulines, and Maxim Panov. Monte carlo variational auto-encoders. In *International Conference on Machine Learning*, pages 10247–10257. PMLR, 2021.
- George Tucker, Dieterich Lawson, Shixiang Gu, and Chris J Maddison. Doubly reparameterized gradient estimators for monte carlo objectives. *arXiv preprint arXiv:1810.04152*, 2018.
- Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.
- Christopher Wolf, Maximilian Karl, and Patrick van der Smagt. Variational inference with hamiltonian monte carlo. *arXiv preprint arXiv:1609.08203*, 2016.
- Zuheng Xu and Trevor Campbell. Embracing the chaos: analysis and diagnosis of numerical instability in variational flows. *arXiv preprint arXiv:2307.06957*, 2023.
- Zuheng Xu, Naitong Chen, and Trevor Campbell. Mixflows: principled variational inference via mixed flows. 2023.
- Guodong Zhang, Kyle Hsu, Jianing Li, Chelsea Finn, and Roger B Grosse. Differentiable annealed importance sampling and the perils of gradient noise. *Advances in Neural Information Processing Systems*, 34, 2021.
- David Zoltowski, Diana Cai, and Ryan P Adams. Slice sampling reparameterization gradients. *Advances in Neural Information Processing Systems*, 34:23532–23544, 2021.

Checklist

1. For all models and algorithms presented, check if you include:
 - (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. **[Yes]** The assumptions of our gradient estimator are described in Prop. 1. Detailed algorithm descriptions include Algorithm 1, Algorithm 2, and Algorithm 3 as well as the surrogate ELBO objective in Eqn. 22.
 - (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. **[Yes]** Yes see in particular Prop. 2 and Sec. 3.2.1.
 - (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. **[Yes]** We include executable code at https://github.com/martinjankowiak/rvrs_example_code that reproduces the main results for the logistic regression experiment described in Sec. 7.2 and Fig. 4. Upon acceptance we plan to submit our methodology as a pull request to the open source NumPyro GitHub repository.
2. For any theoretical claim, check if you include:
 - (a) Statements of the full set of assumptions of all theoretical results. **[Yes]**
 - (b) Complete proofs of all theoretical results. **[Yes]** Proofs can be found in Sec. A and Sec. B in the supplement.
 - (c) Clear explanations of any assumptions. **[Yes]**
3. For all figures and tables that present empirical results, check if you include:
 - (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). **[Yes]** See Sec. F in the supplement.
 - (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). **[Yes]** See Sec. F in the supplement.
 - (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). **[Yes]**
 - (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). **[Yes]**
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
 - (a) Citations of the creator If your work uses existing assets. **[Yes]** We cite the UCI repository (Asuncion and Newman, 2007).
 - (b) The license information of the assets, if applicable. **[Not Applicable]**
 - (c) New assets either in the supplemental material or as a URL, if applicable. **[Not Applicable]**
 - (d) Information about consent from data providers/curators. **[Not Applicable]**
 - (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. **[Not Applicable]**
5. If you used crowdsourcing or conducted research with human subjects, check if you include:
 - (a) The full text of instructions given to participants and screenshots. **[Not Applicable]**
 - (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. **[Not Applicable]**
 - (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. **[Not Applicable]**

Supplementary material for Reparameterized Variational Rejection Sampling

A Gradient estimator for the parameters of the proposal distribution

A.1 Reparameterized ϕ gradient estimator

The covariance in Eqn. 5 can be converted into a pathwise gradient estimator. To see this consider the ‘‘fundamental pathwise gradient identity’’ (see e.g. Jankowiak and Obermeyer (2018); Mohamed et al. (2020))⁷

$$\mathbb{E}_{q_\phi(\mathbf{z})} [f(\mathbf{z}) \nabla_\phi \log q_\phi(\mathbf{z})] = \mathbb{E}_{q_\phi(\mathbf{z})} \left[\frac{\partial f(\mathbf{z})}{\partial \mathbf{z}} \cdot \nabla_\phi \mathbf{z} \right] \quad (10)$$

where $f(\mathbf{z})$ can depend on ϕ and where $\nabla_\phi \mathbf{z}$ is a velocity field for the parameter ϕ that can be derived via e.g. the reparameterization trick if $q_\phi(\mathbf{z})$ is reparameterizable. Then use Eqn. 10 to derive the identity

$$\mathbb{E}_{r_{\phi, \theta}(\mathbf{z})} [f(\mathbf{z}) \nabla_\phi \log q_\phi(\mathbf{z})] = \mathbb{E}_{q_\phi(\mathbf{z})} \left[\frac{a_{\theta, \phi}(\mathbf{z})}{\mathcal{Z}_r} f(\mathbf{z}) \nabla_\phi \log q_\phi(\mathbf{z}) \right] \quad (11)$$

$$= \mathbb{E}_{q_\phi(\mathbf{z})} \left[\frac{\partial}{\partial \mathbf{z}} \left(\frac{a_{\theta, \phi}(\mathbf{z})}{\mathcal{Z}_r} f(\mathbf{z}) \right) \cdot \nabla_\phi \mathbf{z} \right] \quad (12)$$

$$= \mathbb{E}_{q_\phi(\mathbf{z})} \left[\frac{1}{\mathcal{Z}_r} \frac{\partial}{\partial \mathbf{z}} (a_{\theta, \phi}(\mathbf{z}) f(\mathbf{z})) \cdot \nabla_\phi \mathbf{z} \right] \quad (13)$$

$$= \mathbb{E}_{q_\phi(\mathbf{z})} \left[\frac{a_{\theta, \phi}(\mathbf{z})}{\mathcal{Z}_r} \left(f(\mathbf{z}) \frac{\partial \log a_{\theta, \phi}(\mathbf{z})}{\partial \mathbf{z}} + \frac{\partial f(\mathbf{z})}{\partial \mathbf{z}} \right) \cdot \nabla_\phi \mathbf{z} \right] \quad (14)$$

$$= \mathbb{E}_{r_{\phi, \theta}(\mathbf{z})} \left[\left(f(\mathbf{z}) \frac{\partial \log a_{\theta, \phi}(\mathbf{z})}{\partial \mathbf{z}} + \frac{\partial f(\mathbf{z})}{\partial \mathbf{z}} \right) \cdot \nabla_\phi \mathbf{z} \right] \quad (15)$$

If we make the substitution $f(\mathbf{z}) \rightarrow g(\mathbf{z})a_{\theta, \phi}(\mathbf{z})$ in Eqn. 15 this identity can be re-expressed as

$$\mathbb{E}_{r_{\phi, \theta}(\mathbf{z})} [g(\mathbf{z})a_{\theta, \phi}(\mathbf{z}) \nabla_\phi \log q_\phi(\mathbf{z})] = \mathbb{E}_{r_{\phi, \theta}(\mathbf{z})} \left[\left(2g(\mathbf{z}) \frac{\partial a_{\theta, \phi}(\mathbf{z})}{\partial \mathbf{z}} + a_{\theta, \phi}(\mathbf{z}) \frac{\partial g(\mathbf{z})}{\partial \mathbf{z}} \right) \cdot \nabla_\phi \mathbf{z} \right] \quad (16)$$

Using the final form of the identity Eqn. 16 we can rewrite Eqn. 5 as follows:

$$\begin{aligned} \nabla_\phi \text{ELBO} &= \text{COV}_{r_{\phi, \theta}(\mathbf{z})} [\mathcal{A}(\mathbf{z}), a_{\theta, \phi}(\mathbf{z}) \nabla_\phi \log q_\phi(\mathbf{z})] \\ &= \mathbb{E}_{r_{\phi, \theta}(\mathbf{z})} [\bar{\mathcal{A}}(\mathbf{z}) a_{\theta, \phi}(\mathbf{z}) \nabla_\phi \log q_\phi(\mathbf{z})] \\ &= \mathbb{E}_{r_{\phi, \theta}(\mathbf{z})} \left[\left(2\bar{\mathcal{A}}(\mathbf{z}) \frac{\partial a_{\theta, \phi}(\mathbf{z})}{\partial \mathbf{z}} + a_{\theta, \phi}(\mathbf{z}) \frac{\partial \mathcal{A}(\mathbf{z})}{\partial \mathbf{z}} \right) \cdot \nabla_\phi \mathbf{z} \right] \end{aligned} \quad (17)$$

where we have defined

$$\bar{\mathcal{A}}(\mathbf{z}) \equiv \mathcal{A}(\mathbf{z}) - \mathbb{E}_{r_{\phi, \theta}(\mathbf{z}')} [\mathcal{A}(\mathbf{z}')] \quad (18)$$

and used that

$$\frac{\partial}{\partial \mathbf{z}} \bar{\mathcal{A}}(\mathbf{z}) = \frac{\partial}{\partial \mathbf{z}} \mathcal{A}(\mathbf{z}) \quad (19)$$

We also note that Eqn. 17 can be expressed in covariance form as follows (although we prefer the more compact form utilizing $\bar{\mathcal{A}}(\mathbf{z})$):

$$\nabla_\phi \text{ELBO} = \text{COV}_{r_{\phi, \theta}(\mathbf{z})} \left[2\mathcal{A}(\mathbf{z}), \frac{\partial a_{\theta, \phi}(\mathbf{z})}{\partial \mathbf{z}} \cdot \nabla_\phi \mathbf{z} \right] + \mathbb{E}_{r_{\phi, \theta}(\mathbf{z})} \left[a_{\theta, \phi}(\mathbf{z}) \frac{\partial \mathcal{A}(\mathbf{z})}{\partial \mathbf{z}} \cdot \nabla_\phi \mathbf{z} \right] \quad (20)$$

Finally we note that, as we would expect, Eqn. 17 reduces to the standard reparameterized gradient in the limit that $a_{\theta, \phi}(\mathbf{z}) \rightarrow 1$ and $r_{\phi, \theta}(\mathbf{z}) \rightarrow q_\phi(\mathbf{z})$:

$$\nabla_\phi \text{ELBO} \rightarrow \mathbb{E}_{q_\phi(\mathbf{z})} \left[\frac{\partial \mathcal{A}(\mathbf{z})}{\partial \mathbf{z}} \cdot \nabla_\phi \mathbf{z} \right] = \mathbb{E}_{q_\phi(\mathbf{z})} \left[\frac{\partial}{\partial \mathbf{z}} (\log p_\theta(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z})) \cdot \nabla_\phi \mathbf{z} \right] \quad (21)$$

⁷Note that Eqn. 10 is equal to $\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z})} [f(\mathbf{z})] - \mathbb{E}_{q_\phi(\mathbf{z})} [\nabla_\phi f(\mathbf{z})]$ but this fact is not needed for our derivation.

A.2 Automatic differentiation and Monte Carlo details for ELBO and gradient estimation

To get unbiased estimates of Eqn. 8 we need⁸ to draw $S > 1$ samples simultaneously, i.e. just like VRS RVRS utilizes a multi-sample objective. In particular if $\mathbf{z}_s \sim r_{\phi, \theta}$ for $s = 1, \dots, S$ and we use a reparameterized sampler for $q_\phi(\mathbf{z})$ so that \mathbf{z}_s depends explicitly on ϕ according to the automatic differentiation system (e.g. `torch.autograd`), we can define the following surrogate ELBO:

$$\mathcal{L}_{\text{sur}} = \frac{2}{S-1} \sum_{s=1}^S \overbrace{\left\{ \mathcal{A}(\mathbf{z}_s) - \mu_{\mathcal{A}}(\mathbf{z}_{1:S}) \right\}}^{\widetilde{a}_{\theta, \phi}(\mathbf{z}_s)} + \frac{1}{S} \sum_{s=1}^S \overbrace{a_{\theta, \phi}(\mathbf{z}_s)}^{\widetilde{\mathcal{A}}(\mathbf{z}_s)} \quad (22)$$

where $\widetilde{f}(\mathbf{z})$ denotes `stop_gradient(f(z))` and $\check{f}(\mathbf{z})$ denotes `stop_gradient(f)(z)` and

$$\mu_{\mathcal{A}}(\mathbf{z}_{1:S}) \equiv \frac{1}{S} \sum_{s=1}^S \mathcal{A}(\mathbf{z}_s) \quad (23)$$

To derive Eqn. 22 we used the identity

$$\text{COV}_{r(\mathbf{z})}[A(\mathbf{z}), B(\mathbf{z})] \approx \frac{1}{S} \sum_{s=1}^S \left(A(\mathbf{z}_s) - \frac{1}{S-1} \sum_{s' \neq s} A(\mathbf{z}_{s'}) \right) B(\mathbf{z}_s) \quad (24)$$

$$= \frac{1}{S} \sum_{s=1}^S \left(A(\mathbf{z}_s) - \frac{1}{S-1} \left(-A(\mathbf{z}_s) + \sum_{s'=1}^S A(\mathbf{z}_{s'}) \right) \right) B(\mathbf{z}_s) \quad (25)$$

$$= \frac{1}{S} \sum_{s=1}^S \left(\left(1 + \frac{1}{S-1} \right) A(\mathbf{z}_s) - \frac{1}{S-1} \sum_{s'=1}^S A(\mathbf{z}_{s'}) \right) B(\mathbf{z}_s) \quad (26)$$

$$= \frac{1}{S} \sum_{s=1}^S \left(\frac{S}{S-1} A(\mathbf{z}_s) - \frac{S}{S-1} \frac{1}{S} \sum_{s'=1}^S A(\mathbf{z}_{s'}) \right) B(\mathbf{z}_s) \quad (27)$$

$$= \frac{1}{S-1} \sum_{s=1}^S \left(A(\mathbf{z}_s) - \frac{1}{S} \sum_{s'=1}^S A(\mathbf{z}_{s'}) \right) B(\mathbf{z}_s) \quad (28)$$

$$(29)$$

By construction when \mathcal{L}_{sur} in Eqn. 22 is run through autograd we get an unbiased estimate of Eqn. 8. For the purposes of tracking the ELBO for evaluation we get a (biased) MC estimator as follows:

$$\mathcal{L} = \frac{1}{S} \sum_{s=1}^S \mathcal{A}(\mathbf{z}_s) + \log \mathcal{Z}_r \quad (30)$$

where

$$\log \mathcal{Z}_r = \log \mathbb{E}_{q_\phi(\mathbf{z})} [a_{\theta, \phi}(\mathbf{z})] \approx \log \frac{1}{S} \sum_{s=1}^S a_{\theta, \phi}(\mathbf{z}'_s) \quad (31)$$

where $\mathbf{z}'_s \sim q_\phi$ for $s = 1, \dots, S$. In practice we use a large number of samples (e.g. $S \sim 10^4 - 10^5$) to evaluate $\log \mathcal{Z}_r$.

A.3 Runtime considerations

Nothing about VRS or RVRS depends on the specific ansatz for $a_{\phi, \theta}(\mathbf{z})$ in Eqn. 2, apart from the generic requirement (for RVRS) that $a_{\phi, \theta}(\mathbf{z})$ depend on ϕ through $q_\phi(\mathbf{z})$ and that $a_{\phi, \theta}(\mathbf{z}) \in [0, 1]$. We can thus consider

⁸Note that another option would be to keep a running estimate of $\mathbb{E}_{r_{\phi, \theta}(\mathbf{z})} [\mathcal{A}(\mathbf{z})]$ and use this in Eqn. 17 and Eqn. 18. This would result in a biased estimator, but the bias should be minimal given that ϕ and θ change slowly over the course of optimization. This is an interesting option that can reduce computational cost by opening the door to single-sample (i.e. $S = 1$) gradient estimation. While we do not explore this option empirically, we have every reason to expect that it would work well.

other forms of $a_{\phi, \theta}(\mathbf{z})$. One potential problem with $a_{\phi, \theta}(\mathbf{z}) = \sigma(\log p_{\theta}(\mathbf{x}, \mathbf{z}) - \log q_{\phi}(\mathbf{z}) + T)$ is that it can lead to very small acceptance probabilities if T is poorly adapted. Consequently it can be useful to place guardrails that mitigate against this possibility. In the following we consider the simple ansatz

$$a_{\phi, \theta, \epsilon}(\mathbf{z}) = \epsilon + (1 - \epsilon)a_{\phi, \theta}(\mathbf{z}) = \epsilon + (1 - \epsilon)\sigma(\log p_{\theta}(\mathbf{x}, \mathbf{z}) - \log q_{\phi}(\mathbf{z}) + T) \quad (32)$$

where $\epsilon > 0$ is some small fixed constant like $\epsilon = 10^{-3}$ or $\epsilon = 10^{-2}$. With this choice $a_{\phi, \theta, \epsilon}(\mathbf{z}) \in (\epsilon, 1)$ which guarantees that $\mathcal{Z}_r \geq \epsilon$. Although this shouldn't be necessary if sufficient care is taken with T adaptation, we use the ansatz in Eqn. 32 in all our experiments to guard against the possibility of excessive runtimes. Here we describe how this choice modifies Prop. 1.

We begin with the VRS formula

$$\nabla_{\phi} \text{ELBO} = \text{COV}_{r_{\phi, \theta, \epsilon}(\mathbf{z})} [\mathcal{A}(\mathbf{z}), \nabla_{\phi} \log \{q_{\phi}(\mathbf{z})a_{\phi, \theta, \epsilon}(\mathbf{z})\}] \quad (33)$$

In the limit that $\epsilon = 0$ this simplifies to $\text{COV}_{r_{\phi, \theta}(\mathbf{z})} [\mathcal{A}(\mathbf{z}), a_{\phi, \theta}(\mathbf{z})\nabla_{\phi} \log q_{\phi}(\mathbf{z})]$, see Eqn. 66. A bit more algebra is involved if $\epsilon > 0$. Indeed we have

$$\nabla_{\phi} \log a_{\phi, \theta, \epsilon}(\mathbf{z}) = \frac{(1 - \epsilon)\nabla_{\phi} a_{\phi, \theta}(\mathbf{z})}{\epsilon + (1 - \epsilon)a_{\phi, \theta}(\mathbf{z})} = \frac{\nabla_{\phi} \log a_{\phi, \theta}(\mathbf{z})}{\frac{\epsilon}{a_{\phi, \theta}(\mathbf{z})(1 - \epsilon)} + 1} \quad (34)$$

Since $\nabla_{\phi} \log a_{\phi, \theta}(\mathbf{z}) = (a_{\phi, \theta}(\mathbf{z}) - 1)\nabla_{\phi} \log q_{\phi}(\mathbf{z})$ we can write

$$\nabla_{\phi} \log \{q_{\phi}(\mathbf{z})a_{\phi, \theta, \epsilon}(\mathbf{z})\} = \left(1 + \frac{a_{\phi, \theta}(\mathbf{z}) - 1}{\frac{\epsilon}{a_{\phi, \theta}(\mathbf{z})(1 - \epsilon)} + 1}\right) \nabla_{\phi} \log q_{\phi}(\mathbf{z}) \quad (35)$$

$$= \frac{\zeta + a_{\phi, \theta}(\mathbf{z})^2}{\zeta + a_{\phi, \theta}(\mathbf{z})} \nabla_{\phi} \log q_{\phi}(\mathbf{z}) \quad (36)$$

where we have defined $\zeta \equiv \epsilon/(1 - \epsilon)$. Thus we have

$$\nabla_{\phi} \text{ELBO} = \mathbb{E}_{r_{\phi, \theta, \epsilon}(\mathbf{z})} [\bar{\mathcal{A}}(\mathbf{z})\nabla_{\phi} \log \{q_{\phi}(\mathbf{z})a_{\phi, \theta, \epsilon}(\mathbf{z})\}] \quad (37)$$

$$= \mathbb{E}_{r_{\phi, \theta, \epsilon}(\mathbf{z})} \left[\bar{\mathcal{A}}(\mathbf{z}) \frac{\zeta + a_{\phi, \theta}(\mathbf{z})^2}{\zeta + a_{\phi, \theta}(\mathbf{z})} \nabla_{\phi} \log q_{\phi}(\mathbf{z}) \right] \quad (38)$$

We can now appeal to the same logic in Eqn. 11 with

$$f(\mathbf{z}) \rightarrow \bar{\mathcal{A}}(\mathbf{z}) \frac{\zeta + a_{\phi, \theta}(\mathbf{z})^2}{\zeta + a_{\phi, \theta}(\mathbf{z})} \quad (39)$$

to write

$$\nabla_{\phi} \text{ELBO} = \mathbb{E}_{r_{\phi, \theta, \epsilon}(\mathbf{z})} \left[\left(f(\mathbf{z}) \frac{\partial \log a_{\phi, \theta, \epsilon}(\mathbf{z})}{\partial \mathbf{z}} + \frac{\partial f(\mathbf{z})}{\partial \mathbf{z}} \right) \cdot \nabla_{\phi} \mathbf{z} \right] \quad (40)$$

We can then use Eqn. 40 to construct a Monte Carlo surrogate ELBO estimator like in Sec. A.2, though we spare the reader the tedious derivation. The upshot is the following estimator:

$$\begin{aligned} \mathcal{L}_{\text{surr}, \epsilon} &= \frac{1}{S-1} \sum_{s=1}^S \overbrace{\left\{ \mathcal{A}(\mathbf{z}_s) - \mu_{\mathcal{A}}(\mathbf{z}_{1:S}) \right\}}^{\text{}} \left(\overbrace{\frac{\zeta + a_{\phi, \theta}(\mathbf{z}_s)^2}{\zeta + a_{\phi, \theta}(\mathbf{z}_s)} \log a_{\phi, \theta, \epsilon}(\mathbf{z}_s)}^{\text{}} + \frac{\zeta + \overbrace{a_{\phi, \theta}(\mathbf{z}_s)^2}^{\text{}}}{\zeta + \overbrace{a_{\phi, \theta}(\mathbf{z}_s)}^{\text{}}}} \right) + \\ &\quad \frac{1}{S} \sum_{s=1}^S \overbrace{\frac{\zeta + a_{\phi, \theta}(\mathbf{z}_s)^2}{\zeta + a_{\phi, \theta}(\mathbf{z}_s)} \tilde{\mathcal{A}}(\mathbf{z}_s)}^{\text{}} \end{aligned} \quad (41)$$

It is straightforward to check that this reduces to Eqn. 22 when $\epsilon = \zeta = 0$.

B Proof of proposition 2

We want to bound the variational gap Δ between $\log p_{\theta}(\mathbf{x}) \equiv \log \mathbb{E}_{p_{\theta}(\mathbf{z})} [p_{\theta}(\mathbf{x}|\mathbf{z})]$ and the ELBO

$$\Delta = \log p_{\theta}(\mathbf{x}) - \text{ELBO} \quad (42)$$

as a function of T . We work under the assumption that $q_{\phi}(\mathbf{z})$ is sufficiently heavy-tailed so that the ratio $\frac{p_{\theta}(\mathbf{x}, \mathbf{z})}{q_{\phi}(\mathbf{z})}$ is well-behaved (see Eqn. 52 below for the precision condition).

We have

$$\Delta = \text{KL}(r_{\phi, \theta}(\mathbf{z}) || p_{\theta}(\mathbf{z}|\mathbf{x})) = \mathbb{E}_{r_{\phi, \theta}(\mathbf{z})} \left[\log \frac{r_{\phi, \theta}(\mathbf{z})}{p_{\theta}(\mathbf{z}|\mathbf{x})} \right] \geq 0 \quad (43)$$

The KL divergence in Eqn. 43 can be decomposed into a positive contribution from where the logarithm is positive and a negative contribution from where the logarithm is negative. Since the KL divergence is non-negative the magnitude of the positive contribution is larger than or equal to the magnitude of the negative contribution. Consequently to bound Δ it suffices to bound the positive contribution.

The ratio in the log in Eqn. 43 is given by

$$\frac{r_{\phi, \theta}(\mathbf{z})}{p_{\theta}(\mathbf{z}|\mathbf{x})} = \frac{q_{\phi}(\mathbf{z})}{\mathcal{Z}_r p_{\theta}(\mathbf{z}|\mathbf{x})} a_{\theta, \phi}(\mathbf{z}) = \frac{q_{\phi}(\mathbf{z})}{\mathcal{Z}_r p_{\theta}(\mathbf{z}|\mathbf{x})} \frac{1}{1 + e^{-T} \frac{q_{\phi}(\mathbf{z})}{\mathcal{Z}_p p_{\theta}(\mathbf{z}|\mathbf{x})}} \quad (44)$$

$$= \frac{1}{\frac{\mathcal{Z}_r p_{\theta}(\mathbf{z}|\mathbf{x})}{q_{\phi}(\mathbf{z})} + e^{-T} \frac{\mathcal{Z}_r}{\mathcal{Z}_p}} = \frac{1}{1 + f(\mathbf{z}|T)} \quad (45)$$

where $\mathcal{Z}_p p_{\theta}(\mathbf{z}|\mathbf{x}) = p_{\theta}(\mathbf{x}, \mathbf{z})$ with $\mathcal{Z}_p \equiv p_{\theta}(\mathbf{x})$ and

$$f(\mathbf{z}|T) \equiv e^{-T} \frac{\mathcal{Z}_r}{\mathcal{Z}_p} - 1 + \mathcal{Z}_r \frac{p_{\theta}(\mathbf{z}|\mathbf{x})}{q_{\phi}(\mathbf{z})} \quad (46)$$

Thus $\log \frac{r_{\phi, \theta}(\mathbf{z})}{p_{\theta}(\mathbf{z}|\mathbf{x})} > 0$ implies that $f(\mathbf{z}|T) < 0$ so that our task is to bound $f(\mathbf{z}|T)$ from below. Since $\mathcal{Z}_r \frac{p_{\theta}(\mathbf{z}|\mathbf{x})}{q_{\phi}(\mathbf{z})} > 0$ we have that

$$f(\mathbf{z}|T) > e^{-T} \frac{\mathcal{Z}_r}{\mathcal{Z}_p} - 1 \quad (47)$$

We compute

$$\mathcal{Z}_r = \mathbb{E}_{q_{\phi}(\mathbf{z})} [a_{\theta, \phi}(\mathbf{z})] = \mathbb{E}_{q_{\phi}(\mathbf{z})} \left[\frac{1}{1 + e^{-T} \frac{q_{\phi}(\mathbf{z})}{\mathcal{Z}_p p_{\theta}(\mathbf{z}|\mathbf{x})}} \right] \quad (48)$$

so that

$$\frac{e^{-T}}{\mathcal{Z}_p} \mathcal{Z}_r = \mathbb{E}_{q_{\phi}(\mathbf{z})} \left[\frac{\frac{e^{-T}}{\mathcal{Z}_p}}{1 + e^{-T} \frac{q_{\phi}(\mathbf{z})}{\mathcal{Z}_p p_{\theta}(\mathbf{z}|\mathbf{x})}} \right] = \mathbb{E}_{q_{\phi}(\mathbf{z})} \left[\frac{1}{e^T \mathcal{Z}_p + \frac{q_{\phi}(\mathbf{z})}{p_{\theta}(\mathbf{z}|\mathbf{x})}} \right] \quad (49)$$

$$= \mathbb{E}_{q_{\phi}(\mathbf{z})} \left[\frac{1}{\frac{q_{\phi}(\mathbf{z})}{p_{\theta}(\mathbf{z}|\mathbf{x})} \left(1 + e^T \mathcal{Z}_p \frac{p_{\theta}(\mathbf{z}|\mathbf{x})}{q_{\phi}(\mathbf{z})} \right)} \right] = \mathbb{E}_{p_{\theta}(\mathbf{z}|\mathbf{x})} \left[\frac{1}{1 + e^T \mathcal{Z}_p \frac{p_{\theta}(\mathbf{z}|\mathbf{x})}{q_{\phi}(\mathbf{z})}} \right] \quad (50)$$

and therefore

$$\frac{e^{-T}}{\mathcal{Z}_p} \mathcal{Z}_r - 1 = \mathbb{E}_{p_{\theta}(\mathbf{z}|\mathbf{x})} \left[\frac{-e^T \mathcal{Z}_p \frac{p_{\theta}(\mathbf{z}|\mathbf{x})}{q_{\phi}(\mathbf{z})}}{1 + e^T \mathcal{Z}_p \frac{p_{\theta}(\mathbf{z}|\mathbf{x})}{q_{\phi}(\mathbf{z})}} \right] > \mathbb{E}_{p_{\theta}(\mathbf{z}|\mathbf{x})} \left[-e^T \mathcal{Z}_p \frac{p_{\theta}(\mathbf{z}|\mathbf{x})}{q_{\phi}(\mathbf{z})} \right] = -e^T \xi \quad (51)$$

where we have defined

$$\xi \equiv \mathbb{E}_{p_{\theta}(\mathbf{z}|\mathbf{x})} \left[\frac{p_{\theta}(\mathbf{x}, \mathbf{z})}{q_{\phi}(\mathbf{z})} \right] = \mathbb{E}_{p_{\theta}(\mathbf{z}|\mathbf{x})} \left[\frac{\mathcal{Z}_p p_{\theta}(\mathbf{z}|\mathbf{x})}{q_{\phi}(\mathbf{z})} \right] > 0 \quad (52)$$

which is finite by assumption so that we can conclude

$$f(\mathbf{z}|T) > -e^T \xi \quad (53)$$

Since

$$\log \frac{r_{\phi, \theta}(\mathbf{z})}{p_{\theta}(\mathbf{z}|\mathbf{x})} = -\log(1 + f(\mathbf{z}|T)) \quad (54)$$

and

$$-\log(1 - x) \leq \frac{3}{2}x \quad \text{for} \quad 0 \leq x \leq \frac{1}{2} \quad (55)$$

we conclude that

$$\log \frac{r_{\phi, \theta}(\mathbf{z})}{p_{\theta}(\mathbf{z}|\mathbf{x})} < \frac{3}{2}e^T \xi \quad \text{for } \mathbf{z} \text{ such that} \quad \log \frac{r_{\phi, \theta}(\mathbf{z})}{p_{\theta}(\mathbf{z}|\mathbf{x})} \geq 0 \quad \text{and} \quad T < -\log 2\xi \quad (56)$$

and consequently

$$\Delta < \frac{3}{2}e^T \xi \quad \text{for} \quad T < -\log 2\xi \quad (57)$$

Since $e^T \rightarrow 0$ as $T \rightarrow -\infty$ we conclude that the variational gap can be made arbitrarily tight. Of course the acceptance probability also goes to zero as $\sim e^T$ in this limit so it becomes increasingly expensive to tighten the gap.

B.1 Semi-RVRS: models with global and local latent variables

Instead of considering generic unstructured models as above, we now consider the scenario introduced in Sec. 4.4, i.e. we consider models with both a global latent variable \mathbf{z}_G and local latent variables $\{\mathbf{z}_n\}$, where $n = 1, \dots, N$ indexes the N observed data points $\{\mathbf{x}_n\}$. (See Sec. E for additional algorithmic details on Semi-RVRS). We assume the following conditional independence structure:

$$p_{\theta}(\mathbf{x}_{1:N}, \mathbf{z}_G, \mathbf{z}_{1:N}) = p_{\theta}(\mathbf{z}_G) \prod_{n=1}^N p_{\theta}(\mathbf{x}_n | \mathbf{z}_n, \mathbf{z}_G) p_{\theta}(\mathbf{z}_n | \mathbf{z}_G) \quad (58)$$

We want to upper bound the variational gap, which is given by

$$\Delta = \text{KL} \left(q_{\phi}(\mathbf{z}_G) \prod_n r_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G) \parallel p_{\theta}(\mathbf{z}_G | \mathbf{x}_{1:N}) \prod_n p_{\theta}(\mathbf{z}_n | \mathbf{z}_G, \mathbf{x}_n) \right) \quad (59)$$

where we have exploited the assumed conditional independence structure to factorize the posterior. We now appeal to the chain rule of KL divergences which reads

$$\text{KL}(q(a, b) \parallel p(a, b)) = \text{KL}(q(a) \parallel p(a)) + \mathbb{E}_{q(a)} [\text{KL}(q(b|a) \parallel p(b|a))] \quad (60)$$

to obtain

$$\begin{aligned} \Delta &= \text{KL} \left(q_{\phi}(\mathbf{z}_G) \parallel p_{\theta}(\mathbf{z}_G | \mathbf{x}_{1:N}) \right) + \mathbb{E}_{q_{\phi}(\mathbf{z}_G)} \left[\text{KL} \left(\prod_n r_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G) \parallel \prod_n p_{\theta}(\mathbf{z}_n | \mathbf{z}_G, \mathbf{x}_n) \right) \right] \\ &= \text{KL} \left(q_{\phi}(\mathbf{z}_G) \parallel p_{\theta}(\mathbf{z}_G | \mathbf{x}_{1:N}) \right) + \sum_n \mathbb{E}_{q_{\phi}(\mathbf{z}_G)} \left[\text{KL} \left(r_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G) \parallel p_{\theta}(\mathbf{z}_n | \mathbf{z}_G, \mathbf{x}_n) \right) \right] \end{aligned} \quad (61)$$

Note that each \mathbf{z}_n KL divergence in Eqn. 61 is precisely equal to the variational gap of a RVRS variational distribution targeting the distribution $p_{\theta}(\mathbf{z}_n|\mathbf{z}_G, \mathbf{x}_n)$ so we can apply the same bounding logic as above (in particular exploiting the linearity in x of the inequality in Eqn. 55) to each latent variable \mathbf{z}_n and obtain the following bound on the variational gap

$$\Delta < \frac{3}{2}e^T \sum_{n=1}^N \xi_n + \text{KL} \left(q_{\phi}(\mathbf{z}_G) \parallel p_{\theta}(\mathbf{z}_G|\mathbf{x}_{1:N}) \right) \quad (62)$$

which is valid for $T < -\log 2 \max_n \xi_n$ where we assume that $T_n = T \forall n$ and we define

$$\xi_n \equiv \mathbb{E}_{q_{\phi}(\mathbf{z}_G)} \mathbb{E}_{p_{\theta}(\mathbf{z}_n|\mathbf{z}_G, \mathbf{x}_n)} \left[\frac{p_{\theta}(\mathbf{x}_n|\mathbf{z}_n, \mathbf{z}_G) p_{\theta}(\mathbf{z}_n|\mathbf{z}_G)}{q_{\phi_n}(\mathbf{z}_n)} \right] \quad (63)$$

Evidently this bound is only meaningful if all ξ_n are finite, which will be true if each proposal distribution $q_{\phi_n}(\mathbf{z}_n)$ is sufficiently heavy-tailed.

C Additional discussion of VRS

C.1 Sampling cost

The number of proposal draws $\mathbf{z} \sim q_{\phi}(\cdot)$ generated before a sample is accepted is governed by a geometric distribution with success probability $\mathcal{Z}_r \equiv \int d\mathbf{z} q_{\phi}(\mathbf{z}) a_{\phi, \theta}(\mathbf{z})$:

$$\text{Prob}(t^{\text{th}} \text{ sample accepted}) = \mathcal{Z}_r (1 - \mathcal{Z}_r)^{t-1} \quad \text{with} \quad t = 1, 2, \dots \quad (64)$$

Since the expected value of a geometric random variable is given by the reciprocal of the success probability, the expected number of draws from the proposal distribution is given by \mathcal{Z}_r^{-1} . Evidently, rejection sampling becomes expensive for small \mathcal{Z}_r .

That the logic behind Eqn. 64 is correct can be corroborated by using the same logic to compute the variational density $r_{\phi, \theta}(\mathbf{z})$ in terms of a geometric series:

$$\begin{aligned} r_{\phi, \theta}(\mathbf{z}) &= \sum_{t=1}^{\infty} \text{Prob} \left(\text{accept } \mathbf{z} \text{ at sampling step } t \mid \text{rejected previous } t-1 \text{ samples} \right) \text{Prob}(\text{reject } t-1 \text{ samples}) \\ &= \sum_{t=1}^{\infty} q_{\phi}(\mathbf{z}) a_{\phi, \theta}(\mathbf{z}) \left(1 - \int q_{\phi}(\mathbf{z}') a_{\phi, \theta}(\mathbf{z}') d\mathbf{z}' \right)^{t-1} \\ &= q_{\phi}(\mathbf{z}) a_{\phi, \theta}(\mathbf{z}) \sum_{t=0}^{\infty} (1 - \mathcal{Z}_r)^t = q_{\phi}(\mathbf{z}) a_{\phi, \theta}(\mathbf{z}) \frac{1}{1 - (1 - \mathcal{Z}_r)} = \frac{q_{\phi}(\mathbf{z}) a_{\phi, \theta}(\mathbf{z})}{\mathcal{Z}_r} \end{aligned} \quad (65)$$

See Bauer and Mnih (2019) for an analogous derivation.

C.2 Gradient estimators

The gradient estimator for proposal parameters ϕ for the VRS ELBO can be expressed in a number of equivalent ways

$$\begin{aligned} \nabla_{\phi} \text{ELBO} &= \text{COV}_{r_{\phi, \theta}(\mathbf{z})} [\mathcal{A}(\mathbf{z}), \nabla_{\phi} \log \{q_{\phi}(\mathbf{z}) a_{\phi, \theta}(\mathbf{z})\}] \\ &= \text{COV}_{r_{\phi, \theta}(\mathbf{z})} [\mathcal{A}(\mathbf{z}), (1 - \sigma(\ell_{\theta, \phi}^T(\mathbf{z}))) \nabla_{\phi} \log q_{\phi}(\mathbf{z})] \\ &= \text{COV}_{r_{\phi, \theta}(\mathbf{z})} [\mathcal{A}(\mathbf{z}), \sigma(-\ell_{\theta, \phi}^T(\mathbf{z})) \nabla_{\phi} \log q_{\phi}(\mathbf{z})] \\ &= \text{COV}_{r_{\phi, \theta}(\mathbf{z})} [\mathcal{A}(\mathbf{z}), a_{\phi, \theta}(\mathbf{z}) \nabla_{\phi} \log q_{\phi}(\mathbf{z})] \end{aligned} \quad (66)$$

where $\mathcal{A}(\mathbf{z}) \equiv \log p_{\theta}(\mathbf{x}, \mathbf{z}) - \log q_{\phi}(\mathbf{z}) - \log a_{\phi, \theta}(\mathbf{z})$. In the limit that $T \rightarrow \infty$ we have $a_{\phi, \theta}(\mathbf{z}) \rightarrow 1$ and $r_{\phi, \theta}(\mathbf{z}) \rightarrow q_{\phi}(\mathbf{z})$. Thus in this limit Eqn. 66 becomes

$$\nabla_{\phi} \text{ELBO} \rightarrow \text{COV}_{q_{\phi}(\mathbf{z})} [\log p_{\theta}(\mathbf{x}, \mathbf{z}) - \log q_{\phi}(\mathbf{z}), \nabla_{\phi} \log q_{\phi}(\mathbf{z})] \quad (67)$$

Since we have

$$\int d\mathbf{z} q_\phi(\mathbf{z}) \nabla_\phi \log q_\phi(\mathbf{z}) = \nabla_\phi \int d\mathbf{z} q_\phi(\mathbf{z}) = \nabla_\phi 1 = 0 \quad (68)$$

we can simplify Eqn. 67 further as

$$\nabla_\phi \text{ELBO} \rightarrow \mathbb{E}_{q_\phi(\mathbf{z})} [(\log p_\theta(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z})) \nabla_\phi \log q_\phi(\mathbf{z})] \quad (69)$$

which is precisely the conventional score function (i.e. REINFORCE-like) gradient estimator for the ELBO, used e.g. in (Ranganath et al., 2014). The VRS gradient estimator for model parameters θ can also be expressed in a number of different ways:

$$\begin{aligned} \nabla_\theta \text{ELBO} &= \mathbb{E}_{r_{\phi, \theta}(\mathbf{z})} [\nabla_\theta \log p_\theta(\mathbf{x}, \mathbf{z})] + \text{COV}_{r_{\phi, \theta}(\mathbf{z})} [\mathcal{A}(\mathbf{z}), \nabla_\theta \log a_{\phi, \theta}(\mathbf{z})] \\ &= \mathbb{E}_{r_{\phi, \theta}(\mathbf{z})} [\nabla_\theta \log p_\theta(\mathbf{x}, \mathbf{z})] - \text{COV}_{r_{\phi, \theta}(\mathbf{z})} [\mathcal{A}(\mathbf{z}), \sigma(\ell_{\theta, \phi}^T(\mathbf{z})) \nabla_\theta \log p_\theta(\mathbf{x}, \mathbf{z})] \\ &= \mathbb{E}_{r_{\phi, \theta}(\mathbf{z})} [\nabla_\theta \log p_\theta(\mathbf{x}, \mathbf{z})] - \text{COV}_{r_{\phi, \theta}(\mathbf{z})} [\mathcal{A}(\mathbf{z}), (1 - a_{\phi, \theta}(\mathbf{z})) \nabla_\theta \log p_\theta(\mathbf{x}, \mathbf{z})] \end{aligned} \quad (70)$$

In the limit that $T \rightarrow \infty$ we have

$$\nabla_\theta \text{ELBO} \rightarrow \mathbb{E}_{q_\phi(\mathbf{z})} [\nabla_\theta \log p_\theta(\mathbf{x}, \mathbf{z})] \quad (71)$$

which, as we would expect, is the conventional ELBO gradient estimator for model parameters.

C.3 Monte Carlo Estimation

Due to the covariance terms obtaining unbiased Monte Carlo estimates of the gradient estimators Eqn. 66 and Eqn. 70 requires drawing $S > 1$ samples from $r_{\phi, \theta}(\mathbf{z})$. To do so we appeal to the identity in Eqn. 24. For example we can approximate the ϕ gradient estimator as follows:

$$\nabla_\phi \text{ELBO} = \text{COV}_{r_{\phi, \theta}(\mathbf{z})} [\mathcal{A}(\mathbf{z}), a_{\phi, \theta}(\mathbf{z}) \nabla_\phi \log q_\phi(\mathbf{z})] \quad (72)$$

$$\approx \frac{1}{S-1} \sum_{s=1}^S \left\{ \mathcal{A}(\mathbf{z}_s) - \frac{1}{S} \sum_{s'} \mathcal{A}(\mathbf{z}_{s'}) \right\} a_{\phi, \theta}(\mathbf{z}_s) \nabla_\phi \log q_\phi(\mathbf{z}_s) \quad (73)$$

D Adaptively tuning T

As detailed in Sec. 4.2 we can adjust the rejection threshold T using the gradient

$$\frac{\partial \mathcal{L}}{\partial T} = (\mathcal{Z}_r - \mathcal{Z}_{\text{tgt}}) \mathbb{E}_{q_\phi(\mathbf{z})} \left[\frac{\partial a_{\phi, \theta}(\mathbf{z})}{\partial T} \right] = \mathbb{E}_{q_\phi(\mathbf{z})} [a_{\phi, \theta}(\mathbf{z}) - \mathcal{Z}_{\text{tgt}}] \mathbb{E}_{q_\phi(\mathbf{z})} [a_{\phi, \theta}(\mathbf{z})(1 - a_{\phi, \theta}(\mathbf{z}))]$$

To obtain an unbiased Monte Carlo estimate of this quantity we draw $S > 1$ samples from $r_{\phi, \theta}(\mathbf{z})$ and use the same logic used to derive Eqn. 24 to compute

$$\frac{\partial \mathcal{L}}{\partial T} \approx \widehat{\frac{\partial \mathcal{L}}{\partial T}} = \frac{1}{S} \sum_{s=1}^S \left\{ a_{\phi, \theta}(\mathbf{z}_s)(1 - a_{\phi, \theta}(\mathbf{z}_s)) \left(\frac{1}{S-1} (\sum_{s'=1}^S a_{\phi, \theta}(\mathbf{z}_{s'}) - a_{\phi, \theta}(\mathbf{z}_s)) - \mathcal{Z}_{\text{tgt}} \right) \right\} \quad (74)$$

While this stochastic gradient estimator could be plugged into a variety of optimization algorithms, for simplicity we use vanilla SGD (stochastic gradient descent) with a fixed learning rate of 1. In other words at each step t in RVRS ELBO optimization we make the update

$$T_{t+1} = T_t - \widehat{\frac{\partial \mathcal{L}}{\partial T}} \quad (75)$$

We find that this works well in practice—in particular on all the experiments reported here—although we expect that more sophisticated schemes could perform better. We also note that perfect adaptation of T is not necessary, since—provided T is in the right ballpark—the primary relevance of T is to determine the precise computation to inference fidelity trade-off. For example if we set $\mathcal{Z}_{\text{tgt}} = 0.30$ but end up with $\mathcal{Z}_r = 0.29$ the result is that we used a bit more computation than we intended—and obtained a slightly better variational approximation as a result.

E Semi-RVRS

The variational distribution for Semi-RVRS is given by

$$q_\phi(\mathbf{z}_G) \prod_{n=1}^N r_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G) = \frac{1}{\mathcal{Z}_r} q_\phi(\mathbf{z}_G) \prod_{n=1}^N q_{\phi_n}(\mathbf{z}_n) a_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G) \quad (76)$$

where we assume for simplicity that $q_{\phi_n}(\mathbf{z}_n)$ does not depend explicitly on \mathbf{z}_G (though this could easily be accommodated). Here $q_\phi(\mathbf{z}_G)$ is some reparameterizable and parametric variational distribution and each distribution $r_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G)$ is given by

$$r_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G) \propto q_{\phi_n}(\mathbf{z}_n) a_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G) \quad (77)$$

with

$$a_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G) \equiv \sigma(\log p_\theta(\mathbf{x}_n | \mathbf{z}_n, \mathbf{z}_G) p_\theta(\mathbf{z}_n | \mathbf{z}_G) - \log q_{\phi_n}(\mathbf{z}_n) + T_n) \quad (78)$$

and where each $T_n \in \mathbb{R}$ is a rejection threshold parameter. For details on sampling from Eqn. 76 and ELBO computation see the next section, Sec. E.1. For details on estimating the normalization constant \mathcal{Z}_r for the purposes of evaluation see Sec. E.2.

E.1 ELBO computation and rejection sampling on a parallel machine

The ELBO for Semi-RVRS is given by

$$\mathbb{E}_{q_\phi(\mathbf{z}_G) \prod_n r_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G)} \left[\log p_\theta(\mathbf{z}_G) + \sum_n \log \{ p_\theta(\mathbf{x}_n | \mathbf{z}_n, \mathbf{z}_G) p(\mathbf{z}_n | \mathbf{z}_G) \} - \log q_\phi(\mathbf{z}_G) - \sum_n \log r_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G) \right] \quad (79)$$

To construct Monte Carlo gradient estimates of Eqn. 79 we proceed as follows. First we randomly choose a mini-batch of data of size B specified by unique indices $\{i_1, \dots, i_B\}$ and draw a sample of the global latent variable $\mathbf{z}_G \sim q_\phi(\mathbf{z}_G)$. Next we either run the (potentially slow) unbiased sampler defined in Algorithm 2; otherwise we run the (potentially much faster) biased sampler defined in Algorithm 3. In Algorithm 2 we always return exactly S samples $\mathbf{z}_n^{1:S}$ for each data point n . Since a variable number of proposals may need to be drawn for each data point before this is the case, the runtime of this algorithm can be pretty variable (although this variability can be mitigated by dynamically reallocating compute resources, see Algorithm 2). Since however we have exactly S samples for each data point it is straightforward to follow the recipe in Sec. A.1 to construct an unbiased gradient estimator of the Semi-RVRS ELBO Eqn. 79. If instead we use Algorithm 3 some data points in the mini-batch may have fewer than S accepted samples. Consequently we do not use these data points in constructing our Monte Carlo ELBO gradient estimators (note that we need to appropriately re-scale terms in our Monte Carlo estimator to account for the effectively variable mini-batch size). This introduces some bias, however it makes our Semi-RVRS ELBO gradient estimators quite a bit faster (especially for small \mathcal{Z}_{tgt}), since we do not need to waste compute on ‘stragglers’, i.e. data points that have fewer than S accepted samples. Note that the resulting bias is not expected to be too severe, since the bias is exactly zero if the local acceptance probabilities of each data point are equal (e.g. if they are all exactly equal to \mathcal{Z}_{tgt}). While this condition never holds exactly, it holds approximately if the adaptation of the $\{T_n\}$ is working well, and this is enough to ensure that the bias is minimal provided that S' in Algorithm 3 is sufficiently large so that most data points in each mini-batch (say $> 80 - 90\%$) are accepted. As a rule of thumb one might choose $S' = \text{ceil}(S/\mathcal{Z}_{\text{tgt}})$ or $S' = \text{ceil}(2S/\mathcal{Z}_{\text{tgt}})$. See Fig. 6 in Sec. G for empirical confirmation of this intuition.

Note that the above discussion has focused on the more general case of Semi-RVRS with both global and local latent variables. However the basic logic of Algorithm 2 and Algorithm 3 is also applicable in the case with purely local latent variables: just ignore the global latent variable. Indeed we use Algorithm 3 when training VAEs in Sec. 7.4 and Algorithm 2 when evaluating VAE ELBOs after training.

Algorithm 2 Unbiased sampler for the Semi-RVRS variational distribution in Eqn. 76. The same algorithm can also be used for the case with only local latent variables. Optionally dynamically reallocate compute resources to focus on data points that do not have S accepted samples. **Input:** subsample indices $\{i_1, \dots, i_B\}$, number of samples S per data point, acc. prob. $\{a_{\phi_n, \theta}(\mathbf{z}_n)\}$, and proposals $\{q_{\phi_n}(\mathbf{z}_n)\}$.

```

1: for  $k \leftarrow 1$  to  $B$  do                                ▷ Initialize the number of accepted samples for each data point
2:    $s_k \leftarrow 0$ 
3: end for
4: while  $\min\{s_1, \dots, s_B\} < S$  do
5:   if dynamically reallocating compute then
6:     for  $k \leftarrow 1$  to  $B$  do                            ▷ Compute how many samples are left to draw
7:        $w_k \leftarrow \max\{S - s_k, 0\}$ 
8:     end for
9:   end if
10:  for  $k \leftarrow 1$  to  $B$  do
11:    if dynamically reallocating compute then
12:       $j \sim \text{Categorical}(\frac{w_1}{\sum_m w_m}, \dots, \frac{w_B}{\sum_m w_m})$ 
13:    else
14:       $j \leftarrow k$ 
15:    end if
16:     $n \leftarrow i_j$ 
17:     $\mathbf{z}_n \sim q_{\phi_n}(\mathbf{z}_n)$                                 ▷ Draw from proposal distribution
18:    if  $u < a_{\phi_n, \theta}(\mathbf{z}_n)$  where  $u \sim \text{Uniform}(0, 1)$  then    ▷ Do rejection sampling
19:       $s_j \leftarrow s_j + 1$                                 ▷ Keep track of number of accepted samples for each data point
20:       $\mathbf{z}_n^{s_j} \leftarrow \mathbf{z}_n$ 
21:    end if
22:  end for
23: end while
24: return  $\{\mathbf{z}_{i_1}^{1:S}, \dots, \mathbf{z}_{i_B}^{1:S}\}$                 ▷ Return exactly  $S$  samples for each data point

```

E.2 The Semi-RVRS normalization constant

The normalization constant \mathcal{Z}_r for the Semi-RVRS variational distribution in Eqn. 76 is given by

$$\mathcal{Z}_r \equiv \mathbb{E}_{q_{\phi}(\mathbf{z}_G)} \prod_{n=1}^N \mathbb{E}_{q_{\phi_n}(\mathbf{z}_n)} [a_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G)] \quad (80)$$

To compute the corresponding ELBO for evaluation purposes we need to estimate the quantity $\log \mathcal{Z}_r$, since the ELBO is given by

$$\begin{aligned} \text{ELBO} &= \mathbb{E}_{q_{\phi}(\mathbf{z}_G)} \mathbb{E}_{r_{\phi, \theta}(\mathbf{z}_{1:N} | \mathbf{z}_G)} [\log p_{\theta}(\mathbf{z}_G, \mathbf{z}_{1:N}) - \log r_{\phi, \theta}(\mathbf{z}_{1:N} | \mathbf{z}_G)] \\ &= \mathbb{E}_{q_{\phi}(\mathbf{z}_G)} \mathbb{E}_{r_{\phi, \theta}(\mathbf{z}_{1:N} | \mathbf{z}_G)} [\log p_{\theta}(\mathbf{z}_G, \mathbf{z}_{1:N}) - \log q_{\phi}(\mathbf{z}_{1:N}) - \log a_{\phi, \theta}(\mathbf{z}_{1:N} | \mathbf{z}_G) + \log \mathcal{Z}_r] \end{aligned} \quad (81)$$

where for convenience we write

$$\begin{aligned} r_{\phi, \theta}(\mathbf{z}_{1:N} | \mathbf{z}_G) &= \prod_{n=1}^N r_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G) & q_{\phi}(\mathbf{z}_{1:N}) &= \prod_{n=1}^N q_{\phi_n}(\mathbf{z}_n) \\ a_{\phi, \theta}(\mathbf{z}_{1:N} | \mathbf{z}_G) &= \prod_{n=1}^N a_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G) \end{aligned} \quad (82)$$

Unfortunately it is difficult to construct an unbiased low variance estimator for $\log \mathcal{Z}_r$. Indeed, although the naive plug-in Monte Carlo estimator for Eqn. 80 is consistent, it is biased and is generally expected to be high variance. Consequently *for the purposes of evaluation only*⁹ we replace $\log \mathcal{Z}_r$ with a lower bound that is easier to estimate.

⁹Recall that the ELBO gradient estimators we use, which are based on Prop. 1, are unbiased and low variance.

Algorithm 3 Biased sampler for the Semi-RVRS variational distribution in Eqn. 76. **Input:** subsample indices $\{i_1, \dots, i_B\}$, number of samples S per data point, number of candidates $S' \geq S$, acc. prob. $\{a_{\phi_n, \theta}(\mathbf{z}_n)\}$, and proposals $\{q_{\phi_n}(\mathbf{z}_n)\}$. The same algorithm can also be used for the case with only local latent variables.

```

1: for  $k \leftarrow 1$  to  $B$  do
2:   for  $t \leftarrow 1$  to  $S'$  do
3:      $\mathbf{z}_{i_k}^t \sim q_{\phi_{i_k}}(\mathbf{z}_{i_k})$ 
4:      $u \sim \text{Uniform}(0, 1)$ 
5:      $\text{acc}_k^t \leftarrow u < a_{\phi_{i_k}, \theta}(\mathbf{z}_{i_k})$ 
6:   end for
7:    $\text{j}_{1:S'} \leftarrow \text{argsort}(\text{acc}_k^{1:S'})$  ▷ Acc. samples thus have larger indices than non-acc. samples
8:   for  $t \leftarrow 1$  to  $S'$  do
9:      $\mathbf{z}_{i_k}^t \leftarrow \mathbf{z}_{i_k}^{\text{j}_{S-t+1}}$ 
10:     $\text{acc}_k^t \leftarrow \text{acc}_k^{\text{j}_{S-t+1}}$ 
11:  end for
12:   $\text{mask}_k \leftarrow (\sum_{t=1}^S \text{acc}_k^t = S)$ 
13: end for
14: return  $\{(\mathbf{z}_{i_1}^{1:S}, \text{mask}_1), \dots, (\mathbf{z}_{i_B}^{1:S}, \text{mask}_B)\}$  ▷ Return mask and  $S$  samples for each data point

```

Indeed we just appeal to Jensen’s inequality to obtain

$$\log \mathcal{Z}_r \equiv \log \mathbb{E}_{q_{\phi}(\mathbf{z}_G)} \prod_{n=1}^N \mathbb{E}_{q_{\phi_n}(\mathbf{z}_n)} [a_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G)] \quad (83)$$

$$\geq \mathbb{E}_{q_{\phi}(\mathbf{z}_G)} \log \prod_{n=1}^N \mathbb{E}_{q_{\phi_n}(\mathbf{z}_n)} [a_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G)] \quad (84)$$

$$\equiv \mathcal{L}^{\text{lb}} = \mathbb{E}_{q_{\phi}(\mathbf{z}_G)} \sum_{n=1}^N \log \mathbb{E}_{q_{\phi_n}(\mathbf{z}_n)} [a_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G)] \quad (85)$$

While the plug-in Monte Carlo estimator for \mathcal{L}^{lb} in Eqn. 85 is still biased because the expectations w.r.t. \mathbf{z}_n occur inside of a logarithm, the important point is that \mathcal{L}^{lb} is consistent and low variance. Indeed for local latent variables that are relatively low-dimensional, the plug-in Monte Carlo estimator for $\mathbb{E}_{q_{\phi_n}(\mathbf{z}_n)} [a_{\phi_n, \theta}(\mathbf{z}_n | \mathbf{z}_G)]$ is expected to be low-variance and so the bias will be correspondingly small. As such the use of \mathcal{L}^{lb} in evaluating Semi-RVRS ELBOs is expected to yield high-fidelity low-variance approximations to the exact ELBO, and it is these estimators that we report in our experiment in Sec. 7.5. To be precise we use the following nested Monte Carlo estimator

$$\log \mathcal{Z}_r \approx \frac{1}{M_1} \sum_{m_1=1}^{M_1} \sum_{n=1}^N \log \left\{ \frac{1}{M_2} \sum_{m_2=1}^{M_2} a_{\phi_n, \theta}(\mathbf{z}_{n, m_1, m_2} | \mathbf{z}_{G, m_1}) \right\} \quad (86)$$

with $\mathbf{z}_{G, m_1} \sim q_{\phi}(\cdot)$ and $\mathbf{z}_{n, m_1, m_2} \sim q_{\phi_n}(\cdot)$
for $n = 1, \dots, N$ and $m_1 = 1, \dots, M_1$ and $m_2 = 1, \dots, M_2$

with $M_1 = 10^4$ and $M_2 = 10^3$.

F Experimental details

F.1 General RVRS details

We always use $S = 2$ samples to compute multi-sample RVRS ELBO gradient estimators during training. In all cases we use either mean-field or multivariate¹⁰ Normal proposal distributions $q_{\phi}(\mathbf{z})$. Similar to (Geffner and Domke, 2021) in the context of UHA, we initialize the RVRS proposal distribution with a variational distribution obtained by maximizing a conventional ELBO. We initialize the rejection threshold T to minus the ELBO obtained

¹⁰With Cholesky-parameterized full-rank covariance matrices.

with mean-field variational inference. We use the Adam optimization algorithm for all ELBO optimization Kingma and Ba (2014). For RVRS we use an initial learning rate of 10^{-4} that is decimated twice over the course of training: after 1/3 and 2/3 of the total number of training iterations. Unless specified otherwise we used $\epsilon = 10^{-4}$ (see Eqn. 32).

F.2 Other experimental details

Like RVRS we initialize UHA base distributions with a variational distribution obtained by maximizing a conventional ELBO. For UHA we use an initial learning rate of 10^{-4} that is decimated twice over the course of training: after 1/3 and 2/3 of the total number of training iterations. For UHA we limit the stepsize η to $\eta_{\max} = 0.25$ and initialize step sizes to $\eta = 0.005$. UHA ELBOs are computed using a single sample Monte Carlo estimate during training. For mean-field, IWAE, and flow training we use an initial learning rate of 10^{-3} that is decimated twice over the course of training: after 1/3 and 2/3 of the total number of training iterations. Mean-field and normalizing flow ELBOs are computed using a single sample Monte Carlo estimate during training. Just like for RVRS we use the Adam optimization algorithm for all variational baselines Kingma and Ba (2014). For the Block Neural Autoregressive normalizing flow (De Cao et al., 2020) we use `AutoBNFNormal` implemented in NumPyro with default settings (in particular one layer).

F.3 Datasets

Apart from MNIST we use a number of UCI (Asuncion and Newman, 2007) datasets: MiniBooNE, SUSY, Higgs, Adult, Bank, Mushroom, Thyroid, Spambase, Pol, & Bike.

F.4 Characterizing RVRS

The log density of the non-gaussian target in Fig. 1 is given by the formula

$$\log \phi\left(\frac{x+y}{\sqrt{2}}|0, 1\right) + \log \phi\left(\frac{x-y}{\sqrt{2}}|0, e^{\frac{x+y}{\sqrt{2}}}\right) \quad (87)$$

where $\phi(x|\mu, \sigma^2)$ denotes the density of a Normal distribution with mean μ and variance σ^2 evaluated at x . To train variational approximations we train for 5 million gradient steps. We evaluate ELBOs with 1 million samples and use $\epsilon = 10^{-6}$ for RVRS.

The gradient variance results depicted in Fig. 2 were obtained as follows. We use $N = 100$ data points from the MiniBooNE UCI dataset, which has $D = 51$ covariate dimensions. Additional covariate dimensions are removed (via subsetting the original covariates) or added as needed by sampling i.i.d. from a standard normal distribution. Both VRS and RVRS mean-field gaussian proposal distributions are initialized by optimizing a conventional ELBO for 1000 steps. The threshold T is set to minus the ELBO. Variance estimates are made with 5×10^5 samples.

The results in Fig. 3 were also obtained using $N = 100$ data points from the MiniBooNE UCI dataset. We train for 2.4 million steps and consider \mathcal{Z}_{tgt} ranging from 0.004 to 0.40. See Fig. 9 for additional results pertaining to this experiment.

F.5 Logistic regression

We do a total of 3×10^5 training iterations for the normalizing flow due to its computational cost. For all other methods we do a total of 9×10^5 training iterations. The datasets we use were subsampled down to $N = 100$ training data points. This choice was made to ensure a non-trivial amount of non-gaussianity and to enable a comparison with HMC. 10^5 samples were used for ELBO evaluation for all methods. Timing results are reported using a machine with an AMD EPYC 7R13 CPU.

We used NUTS implemented in NumPyro to generate the samples used to compute Max Slice Wasserstein distances. We used a diagonal mass matrix and 10^4 warmup steps. We generated 5×10^5 post-warmup samples. Every 5th sample was retained for a total of 10^5 samples. We then drew 10^5 independent samples from each variational method. These samples were then used to compute Max Slice Wasserstein distances using POT (Flamary et al., 2021). To compute each Wasserstein distance we use 1000 random projections and average results across 10 replicates.

F.6 Gaussian process classification

We used $N = 256$ data points for training for each dataset. We used a RBF kernel with per-dimension lengthscales and a logistic link function with a Bernoulli likelihood. We trained for 6×10^5 iterations and used 2×10^4 samples for ELBO evaluation. For all methods the base/proposal distribution used is a multivariate Normal distribution with a Cholesky-parameterized full-rank covariance matrix. Due to the delicate linear algebra we do all computations in 64-bit precision.

F.7 Variational autoencoders

For all methods we used the same batch size ($B = 100$), trained for 1500 epochs, and evaluated using 5000 samples. The training/test set consist of 60k/10k images, respectively. The latent variable has a standard Gaussian prior and dimension $D = 50$. Both the encoder and decoder are multilayer perceptrons with two hidden layers of 200 hidden units and with tanh activation functions. All experiments were done on a RTX 2070 GPU with 8GB of memory. We used the Adam optimizer and learning rates were decimated, i.e. reduced by a factor of 10, at 500 and 1000 epochs. When training with a conventional ELBO, IWAE, UHA, and RVRS the initial learning rates were 10^{-3} , 10^{-3} , 10^{-4} , and 10^{-4} , respectively. For both UHA and RVRS encoder-decoder parameters were initialized using the final optimized parameters obtained after training with a conventional ELBO. In UHA we used the same set of (learned) step sizes and mass matrices for all data points, i.e. only the base distribution is amortized. In RVRS we used the biased sampler Algorithm 3 with $S = 2$ and $S' = \text{round}(S/\mathcal{Z}_{\text{tgt}})$ for training. Evaluation was done with Algorithm 2.

We initialize the threshold parameter T_n in RVRS for each training data point to a 50-sample Monte Carlo estimate of its corresponding negative ELBO (obtained with the mean field proposal $q_\phi(\mathbf{z})$). Since we do not amortize T_n , after training we need to choose T_n for each unseen test data point such that the acceptance probability of the rejection sampler will approximately equal \mathcal{Z}_{tgt} . Hence for each test data point, we draw 50 samples $\{\mathbf{z}_n\}$ from the proposal distribution q_ϕ and choose $\{T_n\}$ to minimize the objective $\mathcal{L}(T_n) = \frac{1}{2}(\mathcal{Z}_{r,n} - \mathcal{Z}_{\text{tgt}})^2$ for each data point.

F.8 Hierarchical modeling

Both datasets we use have 5000 data points. We add additional Normally distributed noise to 25% of the data points to drive the model into a regime where the Student’s t likelihood is needed to model the resulting heavy-tailed noise. We use 6×10^5 training iterations for all methods and a mini-batch size of 256. Due to the special functions involved in the Gamma probability density function we do all computations in 64-bit precision.

G Additional experimental results

In Fig. 6 and Fig. 7 we report additional results pertaining to the experiment in Sec. 7.5. In Fig. 8 we compare the training dynamics of VRS and RVRS. In Fig. 9 we explore the performance of our T adaptation scheme. In Fig. 10 we report times per gradient step for the GP experiment in Sec. 7.3. In Table 3 we report additional results for the VAE experiment in Sec. 7.4.

Method	Standard VAE	IWAE-10	IWAE-20	IWAE-40	UHA-10	UHA-20	RVRS-0.1	RVRS-0.05	RVRS-0.025
Train –ELBO	92.00 \pm 0.10	88.66 \pm 0.08	87.93 \pm 0.03	87.35 \pm 0.08	87.01 \pm 0.08	86.04 \pm 0.33	87.58 \pm 0.08	87.10 \pm 0.07	86.87 \pm 0.06
Test –ELBO	95.30 \pm 0.14	91.21 \pm 0.07	90.44 \pm 0.07	89.81 \pm 0.09	89.75 \pm 0.09	88.46 \pm 0.22	90.74 \pm 0.16	90.00 \pm 0.12	89.55 \pm 0.12
ms / grad	0.70	1.06	1.49	1.97	5.17	9.73	1.19	1.36	1.75

Table 3: We report negative ELBO objectives (lower is better; mean \pm standard deviation over 5 replicates) computed on training data and held-out test data together with gradient step times for the VAE experiment in Sec. 7.4. Results obtained with a RTX 2070 GPU. This is the same table as in Table 1 but includes objectives computed on the training set.

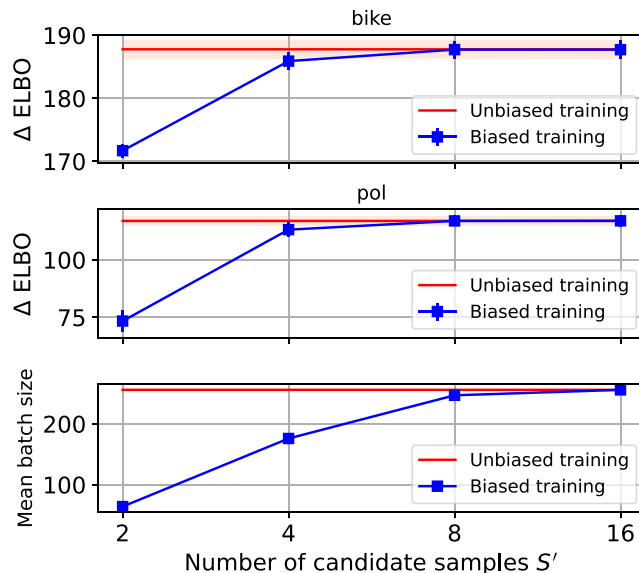


Figure 6: We compare Semi-RVRS training for the hierarchical model in Sec. 7.5 using Algorithm 3 (in blue) and Algorithm 2 (in red). We consider the same two datasets: bike and pol. Uncertainty bands/bars denote 90% confidence intervals obtained from 5 independent runs. As expected provided S' is sufficiently large so that the mean batch size is a large fraction of $B = 256$, then the bias introduced by ‘dropping stragglers’ is minimal and the performance of Algorithm 3 approaches that of Algorithm 2. See Sec. E for additional discussion.

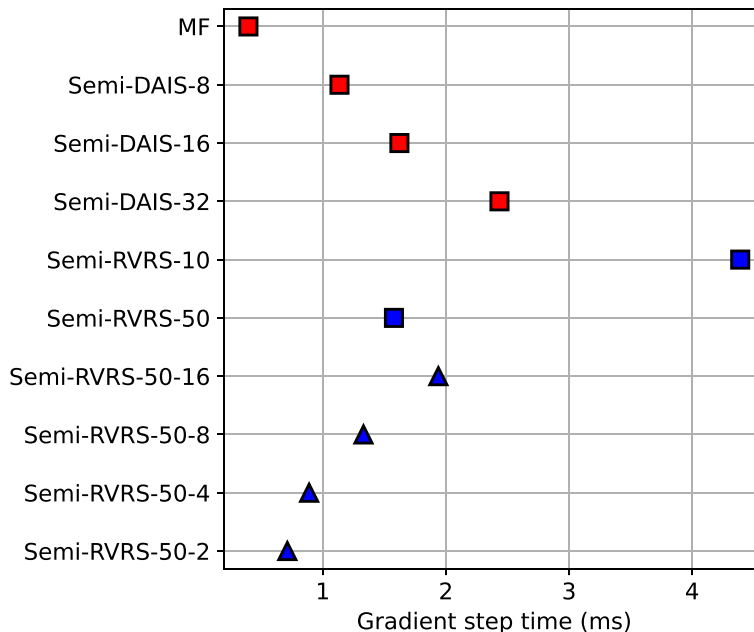


Figure 7: We report training times for the hierarchical model in Sec. 7.5 on the bike dataset. We compare baseline methods (red) to RVRS variants (blue). Among RVRS variants we compare methods using Algorithm 2 (squares) to methods using Algorithm 3 (triangles). Notably Semi-RVRS-0.50-8, i.e. Semi-RVRS with $Z_{tgt} = 0.50$ and $S' = 8$, significantly outperforms e.g. Semi-DAIS-8 (see Table 2 and Fig. 6) but is faster. Timing results are obtained using a machine with an AMD EPYC 7R13 CPU and make it clear that Algorithm 3 can be significantly faster than Algorithm 2 if S' is moderate.

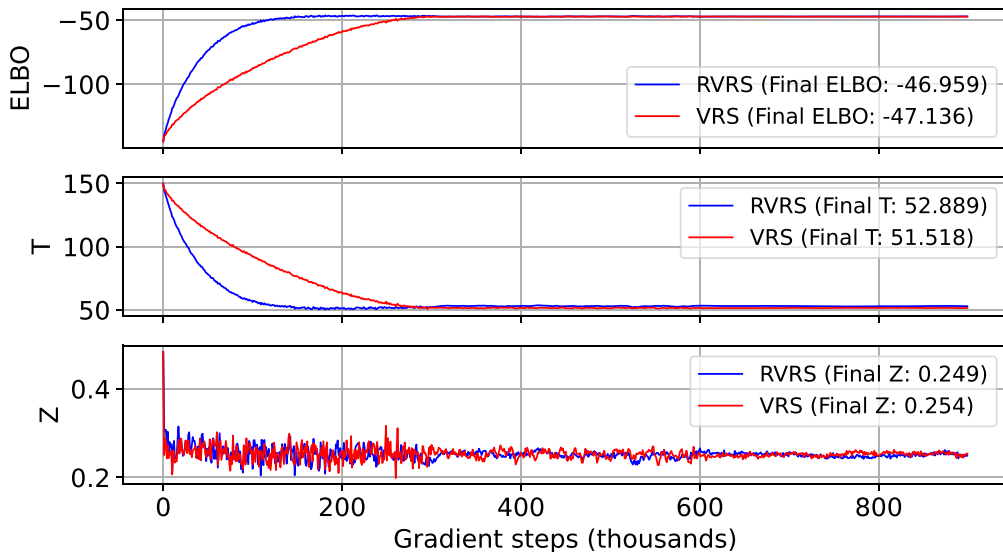


Figure 8: We compare RVRs and VRS training curves for a logistic regression problem with $N = 100$ data points and a $D = 51$ dimensional latent space for $\mathcal{Z}_{\text{tgt}} = 0.25$. From top to bottom we depict the ELBO (computed with 20k samples every 1000 steps), the threshold parameter T , and the value of \mathcal{Z}_r (computed with 20k samples every 1000 steps). The initial learning rate is 10^{-4} and is decimated at 300k and 600k steps. Due to the lower gradient variance of RVRs, RVRs ELBO training makes more rapid progress. For example RVRs attains an ELBO of -50 after ~ 115 k steps, while VRS does not attain this value until ~ 255 k steps. Similarly RVRs attains an ELBO of -100 after ~ 24 k steps, while VRS does not attain this value until ~ 69 k steps.

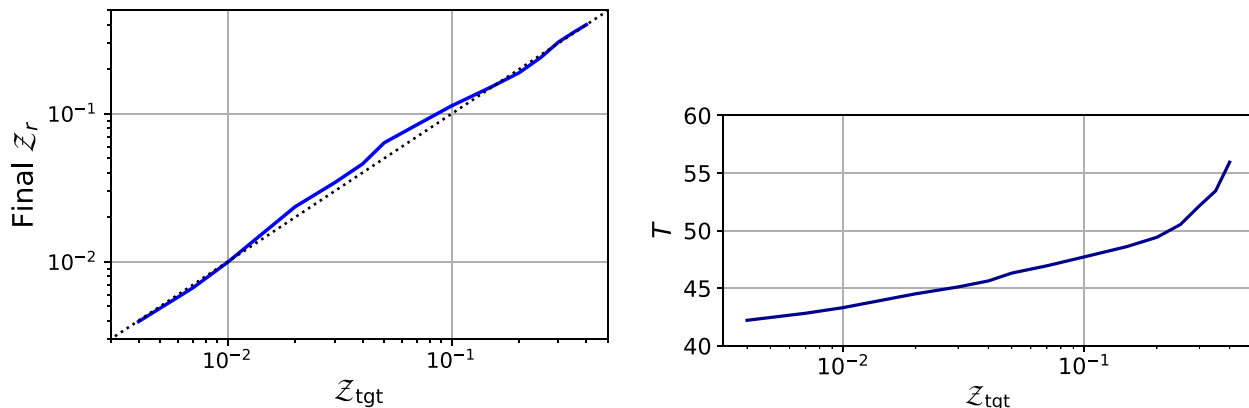


Figure 9: We explore the performance of RVRs as a function of \mathcal{Z}_{tgt} on a logistic regression problem in $D = 51$ dimensions. **(Left)** We show that the T adaptation scheme described in Sec. 4.2 and Sec. D works well over a broad range of \mathcal{Z}_{tgt} . **(Right)** We show how the adapted T changes as a function of \mathcal{Z}_{tgt} . Note that this is a companion figure to Fig. 3.

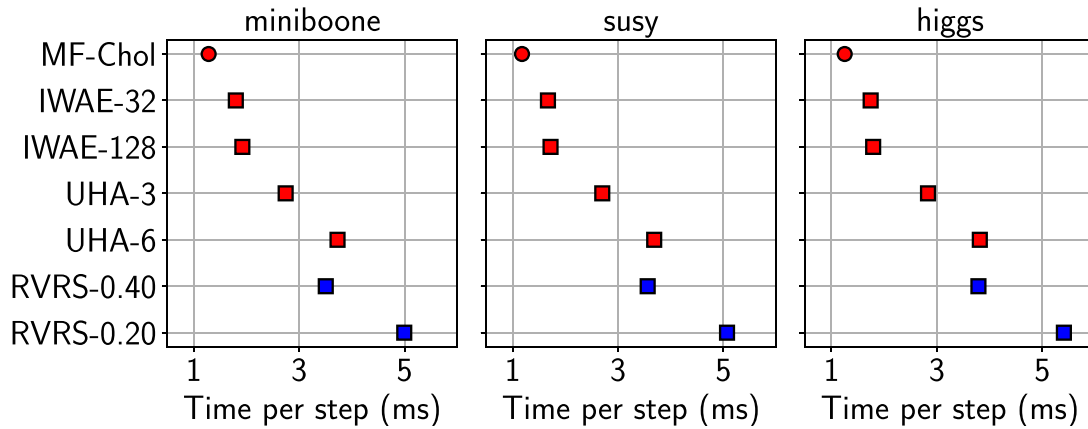


Figure 10: In this companion figure to Fig. 5 we report times per gradient step for the GP classification experiment in Sec. 7.3. Results are obtained with a NVIDIA Tesla V100 GPU. Note that the relative speed of IWAE is a quirk of this particular regime. For relatively moderately sized matrices (here 256×256) commercial GPUs like the V100 can compute a large number of Cholesky decompositions in parallel. As such IWAE parallelizes particularly well in this regime. For larger matrices (e.g. 1024×1024) this advantage would largely evaporate. We also note that we could adapt Algorithm 3 to the setting without local latent variables, which could make RVRS significantly faster when running on parallel-friendly hardware like a GPU.