Gradient Estimation with Discrete Stein Operators

Jiaxin Shi [1] Yuhao Zhou [2] Jessica Hwang [3] Michalis K. Titsias [4] Lester Mackey [1 3]

Abstract

Gradient estimation—approximating the gradient of an expectation with respect to the parameters of a distribution—is central to the solution of many machine learning problems. However, when the distribution is discrete, most common gradient estimators suffer from excessive variance. To improve the quality of gradient estimation, we introduce a variance reduction technique based on Stein operators for discrete distributions. We then use this technique to build flexible control variates for the REINFORCE leave-one-out estimator. Our control variates can be adapted online to minimize the variance and do not require extra evaluations of the target function. In benchmark generative modeling tasks such as training binary variational autoencoders, our gradient estimator achieves substantially lower variance than state-of-the-art estimators with the same number of function evaluations.

1. Introduction

Modern machine learning relies heavily on gradient methods to optimize the parameters of a learning system. However, exact gradient computation is often difficult. For example, in variational inference for training latent variable models (Paisley et al., 2012; Kingma & Welling, 2014) and policy gradient algorithms in reinforcement learning (Williams, 1992) the exact gradient features an intractable sum or integral introduced by an expectation under an evolving probability distribution. To make progress, one resorts to estimating the gradient by drawing samples from that distribution (Mohamed et al., 2020).

The two main classes of gradient estimators used in machine learning are the pathwise or reparameterization gradient estimators (Kingma & Welling, 2014; Rezende et al., 2014; Titsias & Lázaro-Gredilla, 2014) and the REINFORCE or score function estimators (Williams, 1992; Glynn, 1990). The pathwise estimators have shown great success in training variational autoencoders (Kingma & Welling, 2014) but are only applicable to continuous probability distributions. The REINFORCE estimators are more general-purpose and easily accommodate discrete distributions but often suffer from excessively high variance.

To improve the quality of REINFORCE estimators, we develop a new variance reduction technique for discrete expectations. Our method is based upon Stein operators (see, e.g., Stein, 1972; Anastasiou et al., 2021), computable functionals that generate mean-zero functions under a target distribution and provide a natural way of designing control variates for stochastic estimation.

We first provide a general recipe for constructing practical Stein operators for discrete distributions, generalizing the prior literature (Brown & Xia, 2001; Holmes, 2004; Eichelsbacher & Reinert, 2008; Bresler & Nagaraj, 2019; Reinert & Ross, 2019; Yang et al., 2018; Barp et al., 2019). We then develop a gradient estimation framework—RODEO—that augments REINFORCE estimators with mean-zero control variates generated from Stein operators. Finally, inspired by Titsias & Shi (2021), we extend our method to develop control variates for REINFORCE leave-one-out estimators (Salimans & Knowles, 2014; Kool et al., 2019) to further reduce the variance.

The benefits of using Stein operators to construct discrete control variates are twofold. First, the operator structure permits us to learn control variates with a flexible functional form such as those parameterized by neural networks. Second, since our operators are derived from Markov chains on the discrete support, they naturally incorporate information from neighboring states of the process for variance reduction.

We evaluate RODEO on 15 benchmark tasks, including training binary variational autoencoders (VAEs) with one or more stochastic layers. In most cases and with the same number of function evaluations, RODEO delivers lower variance and better training objectives than the state-of-the-art gradient estimators DisARM (Dong et al., 2020; Yin et al., 2020), ARMS (Dimitriev & Zhou, 2021), Double CV (Titsias & Shi, 2021), and RELAX (Grathwohl et al., 2018).

[1]Microsoft Research New England [2]Tsinghua University [3]Stanford University [4]DeepMind. Correspondence to: Jiaxin Shi <jiaxinshi@microsoft.com>.

2. Background

We consider the problem of maximizing the objective function $\mathbb{E}_{q_\eta}[f(x)]$ with respect to the parameters η of a discrete distribution $q_\eta(x)$. Throughout the paper we assume $f(x)$ is a differentiable function of real-valued inputs $x \in \mathbb{R}^d$ but is only evaluated at a discrete subset of the domain due to q_η's being discrete.[1] To maximize $\mathbb{E}_{q_\eta}[f(x)]$ we would ideally compute its gradient as

$$\nabla_\eta \mathbb{E}_{q_\eta}[f(x)] = \mathbb{E}_{q_\eta}[f(x)\nabla_\eta \log q_\eta(x)], \tag{1}$$

where we use the identity $\nabla_\eta q_\eta(x) = q_\eta(x)\nabla_\eta \log q_\eta(x)$ to write the gradient again as an expectation. However, exact computation of (1) is typically intractable due to the complex nature of $f(x)$ and $q_\eta(x)$. The standard workaround is to employ the Monte Carlo gradient estimator known as the score function or REINFORCE estimator (Glynn, 1990; Williams, 1992):

$$\tfrac{1}{K}\textstyle\sum_{k=1}^{K}(f(x_k) - b)\nabla_\eta \log q_\eta(x_k), \quad x_{1:K} \overset{i.i.d.}{\sim} q_\eta. \tag{2}$$

Here, b is a constant called the "baseline" introduced to reduce the variance of the estimator by reducing the scaling effect of $f(x_k)$. Since $\mathbb{E}_{q_\eta}[\nabla_\eta \log q_\eta(x)] = 0$, the REINFORCE estimator is unbiased for any choice of b, and the optimal baseline can be estimated using current or historical function evaluations (Weaver & Tao, 2001; Bengio et al., 2013; Ranganath et al., 2014). A simpler approach which introduces no additional variance is to use moving averages of f from historical evaluations or to train function approximators to mimic those values (Mnih & Gregor, 2014). When $K \geq 2$, a powerful variant of (2) is obtained by replacing b with the leave-one-out average of function values:

$$\tfrac{1}{K}\textstyle\sum_{k=1}^{K}\left(f(x_k) - \tfrac{1}{K-1}\sum_{j\neq k} f(x_j)\right)\nabla_\eta \log q_\eta(x_k). \tag{3}$$

This approach is often called the REINFORCE leave-one-out (RLOO) estimator (Salimans & Knowles, 2014; Kool et al., 2019; Richter et al., 2020) and was recently observed to have very strong performance in training discrete latent variable models (Richter et al., 2020; Dong et al., 2020).

All the above methods construct a baseline that is independent of the "current" sample x_k, but this is not the only way to preserve the unbiasedness of the estimator. We are free to use a sample-dependent baseline b_k as long as the expectation $\mathbb{E}_{q_\eta}[b_k \nabla_\eta \log q_\eta(x_k)]$ is easily computed or has a low-variance unbiased estimate. In this case, we can correct for any bias introduced by b_k by adding in this expectation:

$$\tfrac{1}{K}\textstyle\sum_{k=1}^{K}(f(x_k) - b_k)\nabla_\eta \log q_\eta(x_k) + \mathbb{E}[b_k \nabla_\eta \log q_\eta(x_k)].$$

[1]As detailed by Grathwohl et al. (2021), this assumption holds for a large number of discrete probabilistic models including the binary VAEs of Section 6.

Notably, the sample-dependent baseline b_k can depend not only on the "current" sample x_k but also on the remaining $\{x_j\}_{j\neq k}$. Paisley et al. (2012) suggests choosing b_k as a lower bound of $f(x_k)$ for logistic regression models and a second-order Taylor approximation for more general cases. Double CV (Titsias & Shi, 2021) combines a variance-reduced leave-one-out baseline with a Taylor expansion-like approximation of f to design a highly effective sample-dependent baseline. REBAR (Tucker et al., 2017) constructs the baseline through continuous relaxation of the discrete distribution (Jang et al., 2017; Maddison et al., 2017) and applies reparameterization gradients to the correction term. As a result REBAR requires three evaluations of f for each x_k instead of usual single evaluation (see Appendix A for a detailed explanation).

Unbiased gradient estimators allow us to tune their hyperparameters by directly optimizing the variance. One step beyond this is to learn the whole control variate by minimizing variance. This leads to lower variance when the optimization finds better solutions than our handcrafted ones. The RELAX (Grathwohl et al., 2018) estimator generalizes REBAR by noticing that their continuous relaxation can be replaced with a free-form control variate. However, in order to get strong performance, RELAX still includes the continuous relaxation in their control variate and only adds a small deviation to it. Therefore, RELAX also needs three evaluations of f for each x_k and is usually considered more expensive than other estimators.

In the following two sections, we will introduce a new variance reduction method for REINFORCE-type estimators that 1) enables the learning of flexible control variates parameterized by neural networks (similar to RELAX) but 2) does not assume q_η has a continuous reparameterization and 3) requires only a single evaluation of f for each x_k.

3. Control Variates from Discrete Stein Operators

At the heart of our new estimator is a technique for generating flexible discrete *control variates* (Owen, 2013, Ch. 8), that is for generating a rich class of functions that have known expectations under a given discrete distribution q. One way to achieve this is to identify any discrete-time Markov chain $(X_t)_{t=0}^{\infty}$ with stationary distribution q. Then, the transition matrix P, defined via $P_{xy} = P(X_{t+1} = y|X_t = x)$, satisfies $P^\top q = q$ and hence

$$\mathbb{E}_q[(P - I)h] = 0, \tag{4}$$

for any integrable function h.[2] In other words, for any integrable h, the function $(P - I)h$ is a valid control variate as

[2]We overload our notation so that, for any suitable matrix A, $(Ah)(x) \triangleq \sum_y A_{xy}h(y)$.

it has known mean under q. Moreover, the linear operator $P - I$ is an example of a *Stein operator* (Stein, 1972; Anastasiou et al., 2021) in the sense that it generates mean-zero functions under q. In fact, both Stein et al. (2004) and Dellaportas & Kontoyiannis (2012) developed control variates of the form $(P-I)h$ based on reversible discrete-time Markov chains and linear input functions $h(x) = x_i$.

More generally, Barbour (1988) and Henderson (1997) observed that if we identify a continuous-time Markov chain $(X_t)_{t\geq 0}$ with q as its stationary distribution, then the *generator* A, defined via

$$(Ah)(x) = \lim_{t\to 0} \frac{\mathbb{E}[h(X_t)]-h(x)}{t}, \tag{5}$$

satisfies $A^\top q = 0$ and hence

$$\mathbb{E}_q[Ah] = 0 \tag{6}$$

for all integrable h. Therefore, A is also a Stein operator suitable for generating control variates. Moreover, since any discrete-time chain with transition matrix P can be embedded into a continuous-time chain with transition rate matrix $A = P - I$, this continuous-time construction is strictly more general (Stirzaker, 2005, Ch. 4).

3.1. Discrete Stein operators

We next present several examples of broadly applicable discrete Stein operators that can serve as practical defaults.

Gibbs Stein operator The transition kernel of the random-scan Gibbs sampler with stationary distribution q is

$$P_{xy} = \tfrac{1}{d}\textstyle\sum_{i=1}^d q(y_i|x_{-i})\mathbf{1}(y_{-i} = x_{-i}), \tag{7}$$

where $\mathbf{1}(\cdot)$ is the indicator function (see, e.g., Geyer, 2011). The corresponding Stein operator is $A = P - I$ with

$$\begin{aligned}(Ah)(x) = \tfrac{1}{d}\textstyle\sum_{i=1}^d \big(\textstyle\sum_{\substack{y_i\neq x_i,\\ y_{-i}=x_{-i}}} q(y_i|x_{-i})h(y) \\ + (q(x_i|x_{-i}) - 1)h(x)\big).\end{aligned} \tag{8}$$

In the binary variable setting, (8) recovers the operator used by Bresler & Nagaraj (2019); Reinert & Ross (2019) to bound distances between the stationary distributions of Glauber dynamics Markov chains.

Minimum probability flow Stein operator As the basis for minimum probability flow (MPF) learning, Sohl-Dickstein et al. (2011) studied a continuous-time Markov chain with stationary distribution q and generator

$$A_{xy} = \begin{cases} \left(\frac{q(y)}{q(x)}\right)^{1/2} \mathbf{1}(y \in \mathcal{N}_x) & y \neq x, \\ -\sum_{z\neq x} A_{xz} & y = x, \end{cases} \tag{9}$$

where $\mathcal{N}_x$ denotes the transition neighborhood of x. The neighborhood structure can be very sparse as long as it guarantees that probability can flow between any pair of states given sufficient time. This choice yields

$$(Ah)(x) = \sum_{y\in\mathcal{N}_x, y\neq x} \left(\frac{q(y)}{q(x)}\right)^{1/2} (h(y) - h(x)), \tag{10}$$

an operator discussed but not used in Barp et al. (2019).

Birth-death Stein operator For any finite-cardinality space, we may index the elements of $\mathcal{X} = \{z_0, \ldots, z_{m-1}\}$, let $\texttt{idx}(x)$ return the index of the element that x represents, and define the increment and decrement operators

$$\begin{aligned} \texttt{inc}(x) &= z_{(\texttt{idx}(x)+1) \bmod m} \quad \text{and} \\ \texttt{dec}(x) &= z_{(\texttt{idx}(x)-1) \bmod m}. \end{aligned}$$

On this space, the *birth-death process* (Karlin & McGregor, 1957) with birth rates $b_x = \frac{q(\texttt{inc}(x))}{q(x)}$, death rates $d_x = 1$, and generator

$$A_{xy} = \begin{cases} b_x & \text{if } y = \texttt{inc}(x), \\ d_x & \text{if } y = \texttt{dec}(x), \\ -b_x - d_x & \text{if } y = x, \\ 0 & \text{otherwise}, \end{cases}$$

has q as a stationary distribution, as $A^\top q = 0$. This construction yields the birth-death Stein operator

$$\begin{aligned}(Ag)(x) = b_x(g(\texttt{inc}(x)) - g(x)) - \\ d_x(g(x) - g(\texttt{dec}(x))).\end{aligned} \tag{11}$$

Brown & Xia (2001); Holmes (2004); Eichelsbacher & Reinert (2008) used related operators to characterize convergence to discrete target distributions. Moreover, by substituting $h(x) = g(x) - g(\texttt{inc}(x))$ in (11), we recover the *difference Stein operator* proposed by Yang et al. (2018) without reference to birth-death processes:

$$(Ah)(x) = h(\texttt{dec}(x)) - b_x h(x). \tag{12}$$

4. REINFORCE with Discrete Stein Operators

In the previous section, we introduced discrete Stein operators as ways to construct mean-zero control variates. Now we explain how this idea can be applied to the gradient estimation problem described in Section 2.

Recall that REINFORCE estimates the gradient

$$\nabla_\eta \mathbb{E}_{q_\eta}[f(x)] = \mathbb{E}_{q_\eta}[f(x)\nabla_\eta \log q_\eta(x)]. \tag{13}$$

Due to the existence of $\nabla_\eta \log q_\eta(x)$ as a weighting function, we apply a discrete Stein operator to a vector-valued function $\tilde{h} : \mathcal{X} \to \mathbb{R}^d$ per dimension to construct the following estimator with a mean-zero control variate:

$$\mathbb{E}_{q_\eta}[f(x)\nabla_{\eta_i} \log q_\eta(x) + (A\tilde{h}_i)(x)]. \tag{14}$$

Here $\tilde{h}$ is a surrogate function to be learned such that $A\tilde{h}_i$ will be strongly correlated with $f(x)\nabla_{\eta_i} \log q_\eta(x)$ to reduce its variance. The optimal $\tilde{h}_i$ is given by the solution of Poisson equation

$$\mathbb{E}_{q_\eta}[f\nabla_{\eta_i} \log q_\eta] - f\nabla_{\eta_i} \log q_\eta = A\tilde{h}_i. \tag{15}$$

We could learn a separate $\tilde{h}_i$ per dimension to approximate the solution of (15). However, this poses a difficult optimization problem that requires simultaneously solving d Poisson equations. Instead, we will incorporate knowledge about the structure of the solution to simplify the optimization.

To determine a candidate functional form for $\tilde{h}$, we consider an "optimal" Markov chain which generates i.i.d. samples from q_η. In this case, $(P - I)\tilde{h}_i$ becomes $\mathbb{E}_{q_\eta}[\tilde{h}_i] - \tilde{h}_i$, and the optimal solution is $\tilde{h}_i = f\nabla_{\eta_i} \log q_\eta$. Inspired by this, we consider $\tilde{h}$ of the form

$$\tilde{h}(x) = h(x)\nabla_\eta \log q_\eta(x),$$

where we now only need to learn a scalar-valued function h. Notably, when h exactly equals f and $A = (P - I)$ for any discrete time Markov chain kernel P, then our control variate adjustment amounts to Rao-Blackwellization, as we end up replacing $f(x)\nabla_{\eta_i} \log q_\eta(x)$ with its variance-reduced conditional expectation $P(f\nabla_{\eta_i} \log q_\eta)$.

Based on the above reasoning, it is tempting to let the function h depend on f, e.g., $h(x) = H(f(x))$, where H can be parameterized by a flexible function approximator such as a neural network. However, doing this will significantly increase the cost of the estimator because our Stein operators in Section 3 evaluate h at all neighbors of the current sample. To avoid this problem, we first observe that $\tilde{h}$ can be made sample-specific, i.e., we can use a different $\tilde{h}_k$ for each sample x_k:

$$\tfrac{1}{K}\textstyle\sum_{k=1}^K [f(x_k)\nabla_\eta \log q_\eta(x_k) + (A\tilde{h}_k)(x_k)]. \tag{16}$$

We then consider the following choices of h_k that are informed about the values of f while being cheap to evaluate:

$$h_k(y) = \tfrac{1}{K-1}\textstyle\sum_{j\neq k} H(f(x_j), \nabla f(x_j)^\top (y - x_j)). \tag{17}$$

Here y will take on the values of x_k and its neighbors in the Markov chain. We omit x_k in the sum (17) to ensure that the function h_k is independent of x_k and hence that $\mathbb{E}_{q_\eta}[\frac{1}{K}\sum_{k=1}^K (A\tilde{h}_k)(x_k)] = 0$. Moreover, the control variates based on h_k introduce no additional evaluations of f beyond those required for the usual RLOO estimator. Also, as observed by Titsias & Shi (2021), for many applications such as variational autoencoders (Kingma & Welling, 2014) where f has parameters θ and is learned through gradient-based optimization, $\{\nabla f(x_k)\}_{k=1}^K$ can be obtained "for free" from the same backpropagation to compute the θ gradients.

4.1. RLOO with discrete Stein operators

Building upon the double CV framework of Titsias & Shi (2021), we can combine the above technique with a leave-one-out baseline to further reduce the variance of gradient estimates. We call our final estimator RODEO for **R**EINF**O**RCE with **D**iscrete St**E**in **O**perators:

$$\tfrac{1}{K}\textstyle\sum_{k=1}^K [(f(x_k) - \tfrac{1}{K-1}\sum_{j\neq k}(f(x_j) + (Ah_j)(x_j))) \cdot\nabla_\eta \log q_\eta(x_k) + (A\tilde{h}'_k)(x_k)], \tag{18}$$

where $\tilde{h}'_k(y) = h'_k(y)\nabla_\eta \log q_\eta(y)$ and $\{h_k, h'_k\}_{k=1}^K$ are scalar-valued functions. Here, the main idea is to add two separate control variates for the REINFORCE leave-one-out estimator (3): $(Ah_j)(x_j)$ is a scalar-valued control variate introduced to reduce the variance of the leave-one-out baseline $\frac{1}{K-1}\sum_{j\neq k} f(x_j)$, while $(A\tilde{h}'_k)(x_k)$ acts as a global control variate to further reduce the variance of the gradient estimate. We adopt the aforementioned design (17) of h and $\tilde{h}$ for the two control variates. Despite the similarity, we note that there is a key difference from the double CV estimator (Titsias & Shi, 2021): our global control variate cannot be written as a baseline with the same form as the local control variate. The following proposition (proved in Appendix B) establishes the unbiasedness of our estimator.

Proposition 1. *The RODEO estimator* (18) *is unbiased for* $\nabla_\eta \mathbb{E}_{q_\eta}[f(x)]$ *when each* h_k *is defined as in* (17) *and*

$$h'_k(y) = \tfrac{1}{K-1}\textstyle\sum_{j\neq k} H'(f(x_j), \nabla f(x_j)^\top (y - x_j)). \tag{19}$$

4.2. Hyperparameter optimization

In practice, we let the functions H and H' share a neural network architecture with two output units. Since the estimator is unbiased, we can optimize the parameters γ of the network in an online fashion to minimize the variance of the estimator (similar to Grathwohl et al. (2018)). Specifically, denoting the RODEO gradient estimate by $g_\gamma(x_{1:K})$, then

$$\nabla_\gamma \mathrm{Tr}(\mathrm{Var}(g_\gamma(x_{1:K}))) = 2\mathbb{E}[\nabla_\gamma \|g_\gamma(x_{1:K})\|_2^2]. \tag{20}$$

We use an unbiased estimate of this hyperparameter gradient, $2\nabla_\gamma \|g_\gamma(x_{1:K})\|_2^2$, to update γ after each optimization step of η.

5. Related Work

As we have seen in Section 2, there is a long history of designing control variates for REINFORCE estimators using "baselines" (Weaver & Tao, 2001; Bengio et al., 2013;

Table 1. Training binary latent VAEs with $K = 2, 3, 4$ (except for RELAX which uses 3 evaluations) on MNIST, Fashion-MNIST, and Omniglot. We report the average ELBO (± 1 standard error) on the training set after 1M steps over 5 independent runs. The corresponding average 100-sample bounds on test data are reported in Table 2 of Appendix D.

		Bernoulli Likelihoods			Gaussian Likelihoods		
		MNIST	Fashion-MNIST	Omniglot	MNIST	Fashion-MNIST	Omniglot
$K = 2$	DisARM	-102.75 ± 0.08	-237.68 ± 0.13	-116.50 ± 0.04	668.03 ± 0.61	182.65 ± 0.47	446.61 ± 1.16
	Double CV	-102.14 ± 0.06	-237.55 ± 0.16	-116.39 ± 0.10	676.87 ± 1.18	186.35 ± 0.64	447.65 ± 0.87
	RODEO (Ours)	$\mathbf{-101.89 \pm 0.17}$	$\mathbf{-237.44 \pm 0.09}$	$\mathbf{-115.93 \pm 0.06}$	$\mathbf{681.95 \pm 0.37}$	$\mathbf{191.81 \pm 0.67}$	$\mathbf{454.74 \pm 1.11}$
$K = 3$	ARMS	-100.84 ± 0.14	-237.05 ± 0.12	-115.21 ± 0.07	683.55 ± 1.01	193.07 ± 0.34	457.98 ± 1.03
	Double CV	-100.94 ± 0.09	-237.40 ± 0.11	-115.06 ± 0.12	686.48 ± 0.68	193.93 ± 0.20	457.44 ± 0.79
	RODEO (Ours)	$\mathbf{-100.46 \pm 0.13}$	$\mathbf{-236.88 \pm 0.12}$	$\mathbf{-115.01 \pm 0.05}$	$\mathbf{692.37 \pm 0.39}$	$\mathbf{196.56 \pm 0.42}$	461.87 ± 0.90
RELAX (3 evals)		-101.99 ± 0.04	-237.74 ± 0.12	-115.70 ± 0.08	688.58 ± 0.52	196.38 ± 0.66	$\mathbf{462.23 \pm 0.63}$
$K = 4$	ARMS	-100.09 ± 0.11	-236.87 ± 0.15	-114.68 ± 0.04	688.40 ± 0.45	196.89 ± 0.66	461.33 ± 0.60
	Double CV	-100.03 ± 0.23	-236.93 ± 0.17	-114.75 ± 0.08	691.72 ± 0.83	199.15 ± 0.47	463.48 ± 1.28
	RODEO (Ours)	$\mathbf{-99.61 \pm 0.09}$	$\mathbf{-236.74 \pm 0.17}$	$\mathbf{-114.55 \pm 0.07}$	$\mathbf{694.26 \pm 0.46}$	$\mathbf{201.13 \pm 0.73}$	$\mathbf{466.33 \pm 1.21}$

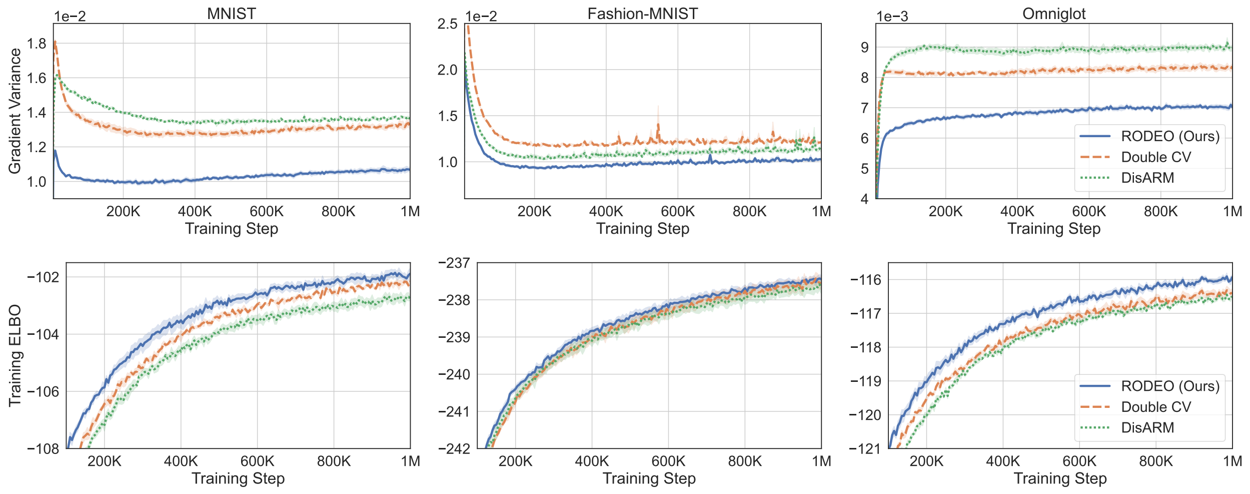

Figure 1. Training binary latent VAEs with $K = 2$ on dynamically binarized MNIST, Fashion-MNIST, and Omniglot. (Top) Variance of gradient estimates. (Bottom) Average ELBO on training examples.

Ranganath et al., 2014; Mnih & Gregor, 2014). The leave-one-out baseline used in Salimans & Knowles (2014); Mnih & Rezende (2016); Kool et al. (2019); Richter et al. (2020) and the sample-dependent baselines in Paisley et al. (2012); Gu et al. (2016); Titsias & Shi (2021); Tucker et al. (2017); Grathwohl et al. (2018) are closely related to our work.

An attractive property of our estimator is that it can incorporate information from neighboring states specified by the Markov chain underlying the Stein operator. As far as we know, there is no other REINFORCE-type estimator that has this property. Estimators based on analytic local expectation (Titsias & Lázaro-Gredilla, 2015; Tokui & Sato, 2017) also use such information. However, they require evaluating the target function in all neighboring states, which is more expensive than evaluating the (cheap) surrogate function as in our method.

Besides control variate methods, prior work has studied other variance reduction methods based on sampling without replacement (Kool et al., 2020) and antithetic sampling (Yin & Zhou, 2019; Dong et al., 2020; Dimitriev & Zhou, 2021; Dong et al., 2021) for gradient estimation. DisARM (Dong et al., 2020; Yin et al., 2020) was shown to outperform RLOO estimators when $K = 2$ and ARMS (Dimitriev & Zhou, 2021) generalizes DisARM to $K > 2$ samples.

Stein operators for continuous distributions have also been leveraged for effective variance reduction in a variety of learning tasks (Assaraf & Caffarel, 1999; Mira et al., 2013; Oates et al., 2017; Liu et al., 2017; Barp et al., 2018; South et al., 2018; Si et al., 2020). Liu et al. (2017) also use Stein operators for gradient estimation. Their method is based on the Langevin Stein operator (Gorham & Mackey, 2015) for continuous distributions, and the resulting estimator coincides with the continuous counterpart of RELAX (Grath-

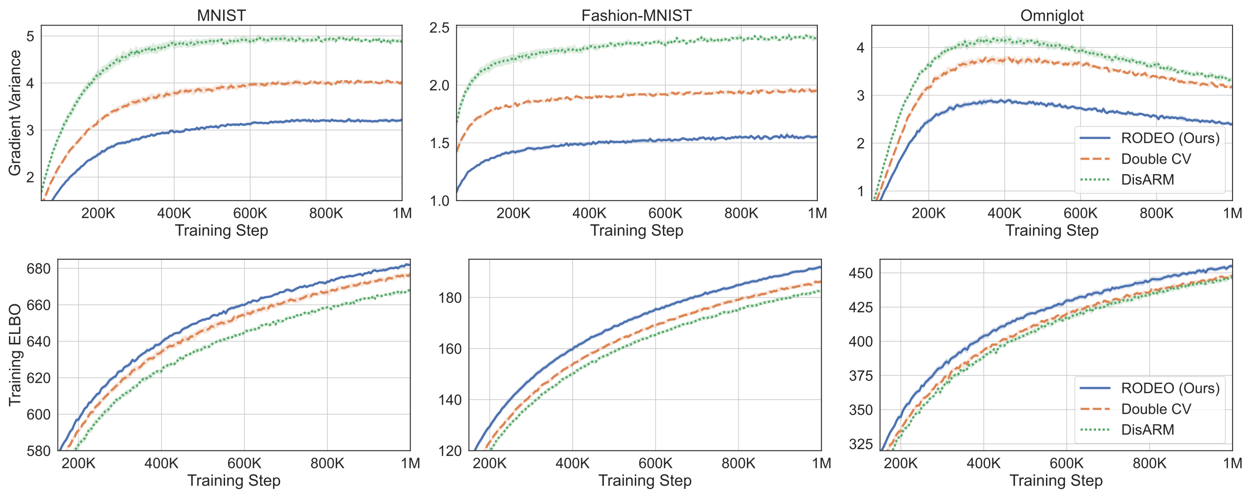

Figure 2. Training binary latent VAEs with Gaussian likelihoods with $K = 2$ on non-binarized MNIST, Fashion-MNIST, and Omniglot. (Top) Variance of gradient estimates. (Bottom) Average ELBO on training examples.

wohl et al., 2018). In contrast, our approach considers discrete Stein operators for Monte Carlo estimation in discrete distributions with exponentially large state spaces.

6. Experiments

Open-source Python code replicating all experiments can be found at `https://github.com/thjashin/rodeo`.

6.1. Training Bernoulli VAEs

Following Dong et al. (2020) and Titsias & Shi (2021), we conduct experiments on training variational auto-encoders (Kingma & Welling, 2014; Rezende et al., 2014) (VAEs) with Bernoulli latent variables. VAEs are models with a joint density $p(y, x) = p(y|x)p(x)$, where x denotes the latent variable. The likelihood $p_\theta(y|x)$ is parameterized by the output of a neural network with x as input and parameters θ. They are typically learned through maximizing the *evidence lower bound* (ELBO):

$$\mathbb{E}_{q_\eta(x|y)}[f(x)], \text{ where } f(x) = \log p_\theta(y, x) - \log q_\eta(x|y),$$

which introduces an inference network $q_\eta(x|y)$. In our experiments, $p(x)$ and $q_\eta(x|y)$ are high-dimensional distributions where each dimension of the random variable is an independent Bernoulli. Since exact gradient computations are intractable, we will use gradient estimators to learn the parameters η of the inference network.

We consider the MNIST (LeCun et al., 1998), Fashion-MNIST (Xiao et al., 2017) and Omniglot (Lake et al., 2015) datasets using their standard train, validation, and test splits. We use both binary and continuous VAE outputs (y) as in Titsias & Shi (2021). In the binary output setting, data are dynamically binarized, and the Bernoulli likelihood is used; in the continuous output setting, data are centered between $[-1, 1]$, and the Gaussian likelihood with learnable diagonal covariance parameters is used.

The VAE architecture and training experimental setup follows Titsias & Shi (2021), except that we decrease the learning rate for binarized Fashion-MNIST to avoid unstable training. Details are given in Appendix C. The dimensionality of the latent variable x is $d = 200$ and x is assigned a uniform factorized Bernoulli prior.

We use the Stein operator derived from the random-scan Gibbs sampler (8) in our estimator. Other choices such as (9) and (12) are examined in Figure 3, and we find that (8) yields better results. The functions H and H' used in (17) and (19) share a neural network architecture with two output units and a single hidden layer consisting of 100 LeakyReLU activation units with coefficient 0.3.

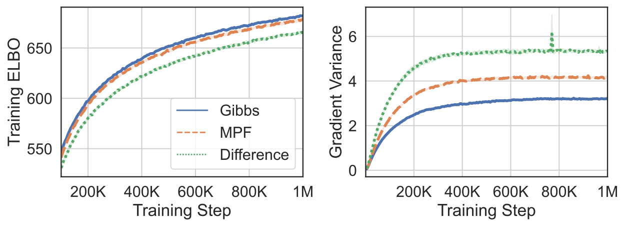

Figure 3. Comparing different Stein operators for training binary latent VAEs: (left) average ELBO on training examples and (right) variance of gradient estimates. Gibbs, MPF and Difference represent the random-scan Gibbs Stein operator (8), the minimum probability flow Stein operator (9) and the difference Stein operator (12), respectively. We used the non-binarized MNIST dataset, Gaussian likelihoods, and $K = 2$ in this experiment.

In the first set of experiments we use two samples, i.e.,

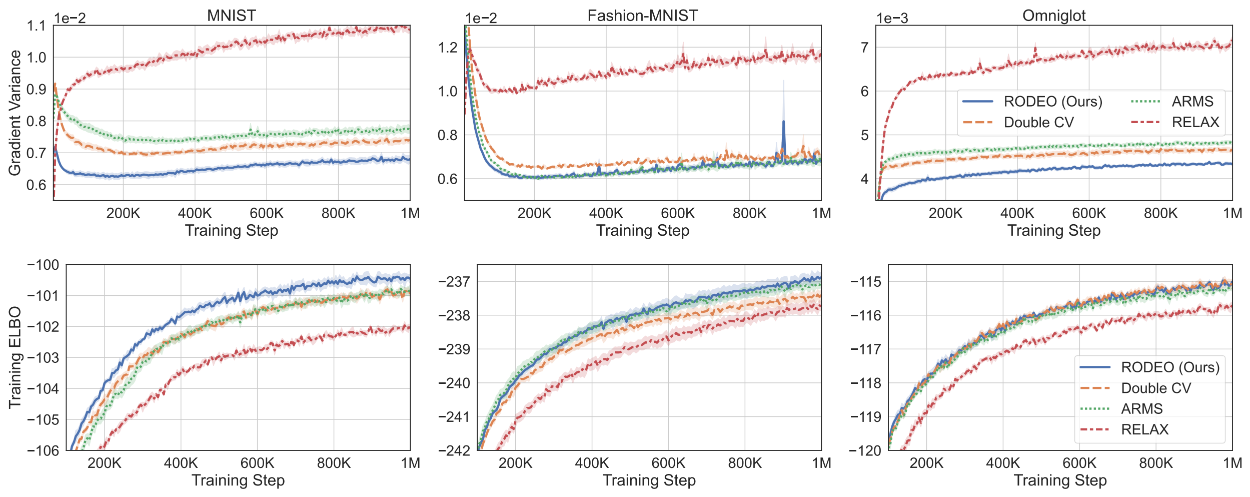

Figure 4. Training binary latent VAEs with three evaluations of f per step using RODEO/Double CV/ARMS with $K = 3$ or RELAX on dynamically binarized MNIST, Fashion-MNIST, and Omniglot. (Top) Variance of gradient estimates. (Bottom) Average ELBO on training examples.

$K = 2$, and compare our variance reduction method to its counterparts including DisARM (Dong et al., 2020) and Double CV (Titsias & Shi, 2021). Results for RLOO are omitted since it is consistently outperformed by Double CV (see Titsias & Shi, 2021). In Figure 1, we plot the average training ELBOs against training steps for all estimators on dynamically binarized datasets. On all three datasets, RODEO achieves the best training ELBOs given the same number of training steps and, in terms of gradient variance, outperforms DisARM and Double CV by a large margin.

Next, we consider VAEs with Gaussian likelihoods trained on non-binarized datasets. In this case the gradient estimates suffer from even higher variance due to the large range of values $f(x)$ can take. The results are plotted in Figure 2. We see that RODEO has substantially lower variance than DisARM and Double CV, leading to significant improvements in training ELBOs. In Figure 6 (Appendix), we plot a tighter 100-sample bound of the marginal log likelihood (Burda et al., 2015) on test examples to better assess the quality of the trained model. RODEO consistently achieves the best test performance in all six settings.

In the second set of experiments we evaluate RODEO, Double CV, and ARMS (Dimitriev & Zhou, 2021) for more than two samples ($K = 3, 4$). All these gradient estimators require a single evaluation of f per sample and hence K evaluations in total per training step. For $K = 3$, we additionally compare with RELAX (Grathwohl et al., 2018) which needs three evaluations of f per step. Figure 4 and Table 1 demonstrate that RODEO outperforms the three previous methods and generally leads to the lowest variance. Although RELAX was often observed to have very strong performance in prior work (Dong et al., 2020; Titsias & Shi, 2021), our results in Figure 4 suggest much larger gains can be achieved by using the same number of function evaluations in other estimators. In practice, one should prefer RODEO over RELAX because f is typically expensive to evaluate.

For all experiments mentioned above, we report the final training ELBOs in Table 1 and test 100-sample bounds in Table 2 (Appendix). We also report the average running time in Table 3. For $K = 2$, RODEO is slightly slower than Double CV and DisARM because it requires $2d$ evaluations of H and H' according to (17). While this results in some extra cost, the network parameterizing H and H' takes only 2-dimensional inputs, produces scalar outputs, and is typically small relative to the VAE model. Hence, our estimator is still much faster than RELAX which needs three evaluation of f. Remarkably, even with one less evaluation of f, RODEO with $K = 2$ achieves comparable or better results than RELAX on dynamically binarized datasets.

As the number of evaluations of H and H' depend quadratically on K when using (17), the cost of these evaluations can dominate for large K; see Table 3. These additional costs are only of value in the situation where we further scale the model size, i.e., where the cost of f is far beyond the cost of H. To obtain a more favorable cost for large K we could consider other choices of the surrogate functions such that the number of neural network evaluations scales linearly with K. For example, one option could be to replace (17) and (19) with

$$\tilde{H}\Big(\tfrac{1}{K-1}\textstyle\sum_{j\neq k} f(x_j), \tfrac{1}{K-1}\sum_{j\neq k} \nabla f(x_j)^\top (y - x_j)\Big).$$

Also, another direction to reduce the extra cost is to choose an appropriate Stein operator to alleviate its dependence

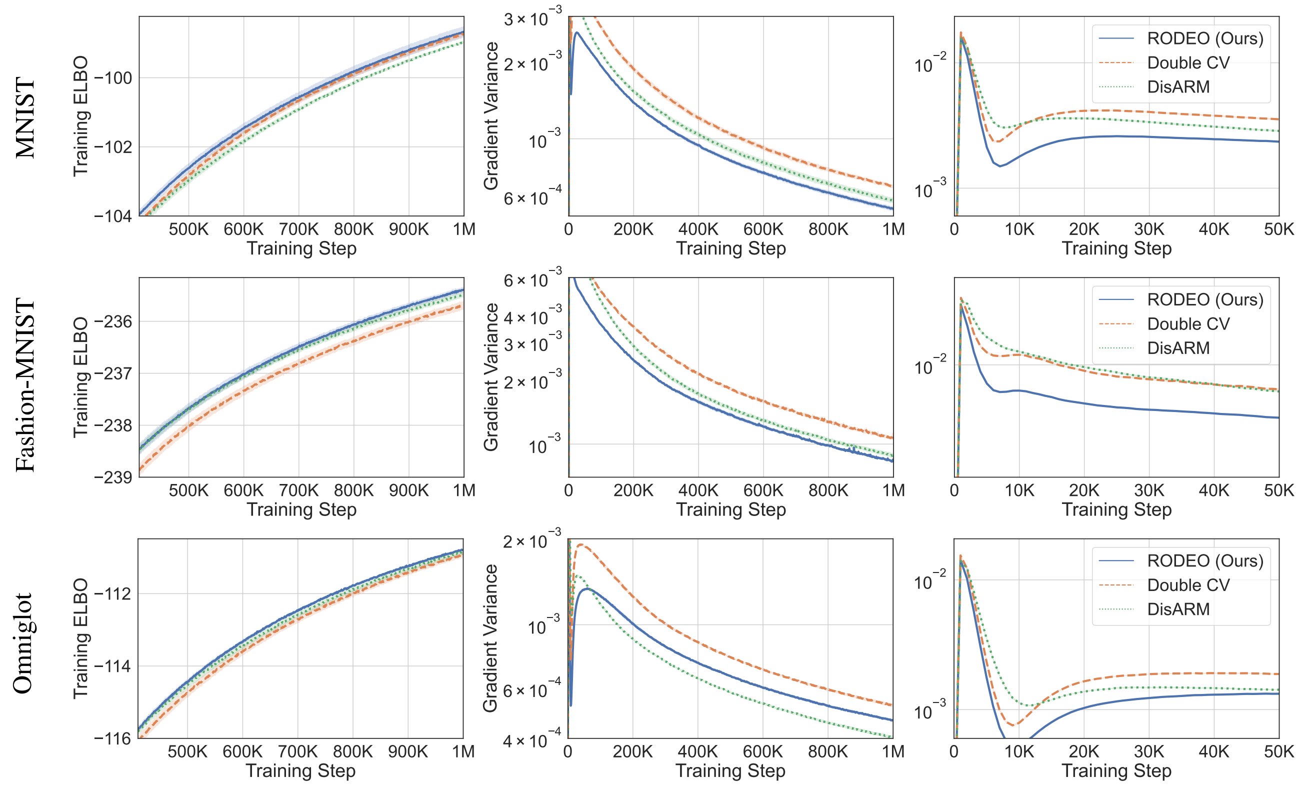

Figure 5. Training hierarchical binary latent VAEs with four stochastic layers. We plot the average ELBO on training examples (left) and variance of gradient estimates (middle). In this experiment, the estimators have very different behaviors towards the beginning and the end of training. We show this on the right figure by zooming into the first 50K steps of the variance plot.

on d. One possible choice is the generator of an ergodic Markov chain with a sparse transition matrix. We leave the exploration of these directions for future work.

6.2. Training hierarchical Bernoulli VAEs

To investigate the performance of RODEO when scaling to hierarchical discrete latent variable models, we follow Dong et al. (2020); Yin & Zhou (2019) to train VAEs with 2/3/4 stochastic layers, each of which consists of 200 Bernoulli variables. We set $K = 2$ and compare our estimator with DisARM and Double CV on dynamically binarized MNIST, Fashion-MNIST, and Omniglot. For each stochastic layer, we use a different control variate network which has the same architecture as those in our VAE experiments from the previous section. More details are presented in Appendix C.

We plot the results on VAEs with four stochastic layers in Figure 5. The results for two and three stochastic layers can be found in Appendix D. RODEO generally achieves the best training ELBOs. One difference we noticed when comparing the variance results with those obtained from single-stochastic-layer VAEs is that these estimators have very different behaviors towards the beginning and the end of training. Double CV starts with lower variance than DisARM, but the gap diminishes after around 100K steps and DisARM start to perform better as the training proceeds. In contrast, RODEO has the lowest variance in both phases with the only exception on Omniglot, where DisARM overtakes it in the long run.

7. Conclusions and Future Work

This work tackles the gradient estimation problem for discrete distributions. We proposed a variance reduction technique which exploits Stein operators to generate control variates for REINFORCE leave-one-out estimators. Our RODEO estimator does not rely on continuous reparameterization of the distribution, requires no additional function evaluations for each sample, and can be adapted online to learn very flexible control variates parameterized by neural networks.

Future work could explore applying this method in the scenarios where the target function is non-differentiable. In order to do this, we must resort to other choices of surrogate functions that use no derivative information. In addition, a systematic study of the relationship between the choice of the Markov chain and performance of the estimator is an interesting topic for future research.

References

Anastasiou, A., Barp, A., Briol, F.-X., Ebner, B., Gaunt, R. E., Ghaderinezhad, F., Gorham, J., Gretton, A., Ley, C., Liu, Q., and Mackey, L. Stein's method meets statistics: A review of some recent developments. *arXiv preprint arXiv:2105.03481*, 2021.

Assaraf, R. and Caffarel, M. Zero-variance principle for Monte Carlo algorithms. *Phys. Rev. Lett.*, 83:4682–4685, Dec 1999.

Barbour, A. D. Stein's method and Poisson process convergence. *J. Appl. Probab.*, (Special Vol. 25A):175–184, 1988. ISSN 0021-9002. A celebration of applied probability.

Barp, A., Oates, C., Porcu, E., and Girolami, M. A Riemann-Stein kernel method. *arXiv preprint arXiv:1810.04946*, 2018.

Barp, A., Briol, F.-X., Duncan, A., Girolami, M., and Mackey, L. Minimum Stein discrepancy estimators. *Advances in Neural Information Processing Systems*, 32: 12964–12976, 2019.

Bengio, Y., Léonard, N., and Courville, A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Bresler, G. and Nagaraj, D. Stein's method for stationary distributions of Markov chains and application to Ising models. *The Annals of Applied Probability*, 29(5):3230–3265, 2019.

Brown, T. C. and Xia, A. Stein's Method and Birth-Death Processes. *The Annals of Probability*, 29(3):1373 – 1403, 2001.

Burda, Y., Grosse, R., and Salakhutdinov, R. Importance weighted autoencoders. *International Conference on Learning Representations*, 2015.

Dellaportas, P. and Kontoyiannis, I. Control variates for estimation based on reversible Markov chain Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 74(1):133–161, 2012.

Dimitriev, A. and Zhou, M. ARMS: Antithetic-REINFORCE-Multi-Sample gradient for binary variables. In *International Conference on Machine Learning*, volume 139, pp. 2717–2727, 2021.

Dong, Z., Mnih, A., and Tucker, G. DisARM: An antithetic gradient estimator for binary latent variables. In *Advances in Neural Information Processing Systems*, volume 33, pp. 18637–18647, 2020.

Dong, Z., Mnih, A., and Tucker, G. Coupled gradient estimators for discrete latent variables. *arXiv preprint arXiv:2106.08056*, 2021.

Eichelsbacher, P. and Reinert, G. Stein's method for discrete gibbs measures. *The Annals of Applied Probability*, 18 (4):1588–1618, 2008.

Geyer, C. J. Introduction to Markov chain Monte Carlo. In Brooks, S., Gelman, A., Jones, G., and Meng, X.-L. (eds.), *Handbook of Markov Chain Monte Carlo*, pp. 3–48. CRC Press, 2011.

Glynn, P. W. Likelihood ratio gradient estimation for stochastic systems. *Commun. ACM*, 33(10):75–84, October 1990.

Gorham, J. and Mackey, L. Measuring sample quality with Stein's method. In *Advances in Neural Information Processing Systems*, pp. 226–234, 2015.

Grathwohl, W., Choi, D., Wu, Y., Roeder, G., and Duvenaud, D. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. In *International Conference on Learning Representations*, 2018.

Grathwohl, W., Swersky, K., Hashemi, M., Duvenaud, D., and Maddison, C. Oops I took a gradient: Scalable sampling for discrete distributions. In *International Conference on Machine Learning*, pp. 3831–3841, 2021.

Gu, S., Levine, S., Sutskever, I., and Mnih, A. MuProp: Unbiased backpropagation for stochastic neural networks. In *International Conference on Learning Representations*, 2016.

Henderson, S. G. *Variance reduction via an approximating Markov process*. PhD thesis, Stanford University, 1997.

Holmes, S. Stein's method for birth and death chains. In Diaconis, P. and Holmes, S. (eds.), *Stein's Method: Expository Lectures and Applications*, pp. 42–65. 2004.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with Gumbel-softmax. *International Conference on Learning Representations*, 2017.

Karlin, S. and McGregor, J. The classification of birth and death processes. *Transactions of the American Mathematical Society*, 86(2):366–400, 1957.

Kingma, D. P. and Welling, M. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

Kool, W., Hoof, H. V., and Welling, M. Buy 4 REINFORCE samples, get a baseline for free! In *DeepRL-StructPred@ICLR*, 2019.

Kool, W., van Hoof, H., and Welling, M. Estimating gradients for discrete random variables by sampling without replacement. In *International Conference on Learning Representations*, 2020.

Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

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

Liu, H., Feng, Y., Mao, Y., Zhou, D., Peng, J., and Liu, Q. Action-depedent control variates for policy optimization via Stein's identity. *arXiv preprint arXiv:1710.11198*, 2017.

Maddison, C. J., Mnih, A., and Teh, Y. W. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

Mira, A., Solgi, R., and Imparato, D. Zero variance Markov chain Monte Carlo for Bayesian estimators. *Statistics and Computing*, 23(5):653–662, 2013.

Mnih, A. and Gregor, K. Neural variational inference and learning in belief networks. In *International Conference on Machine Learning*, pp. 1791–1799, 2014.

Mnih, A. and Rezende, D. Variational inference for Monte Carlo objectives. In *International Conference on Machine Learning*, pp. 2188–2196, 2016.

Mohamed, S., Rosca, M., Figurnov, M., and Mnih, A. Monte Carlo gradient estimation in machine learning. *Journal of Machine Learning Research*, 21(132):1–62, 2020.

Oates, C. J., Girolami, M., and Chopin, N. Control functionals for Monte Carlo integration. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79 (3):695–718, 2017.

Owen, A. B. *Monte Carlo theory, methods and examples*. 2013.

Paisley, J., Blei, D. M., and Jordan, M. I. Variational Bayesian inference with stochastic search. In *International Conference on Machine Learning*, pp. 1363–1370, 2012.

Ranganath, R., Gerrish, S., and Blei, D. Black box variational inference. In *International Conference on Artificial Intelligence and Statistics*, pp. 814–822, 2014.

Reinert, G. and Ross, N. Approximating stationary distributions of fast mixing Glauber dynamics, with applications to exponential random graphs. *The Annals of Applied Probability*, 29(5):3201–3229, 2019.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, 2014.

Richter, L., Boustati, A., Nüsken, N., Ruiz, F., and Akyildiz, O. D. VarGrad: A low-variance gradient estimator for variational inference. In *Advances in Neural Information Processing Systems*, volume 33, pp. 13481–13492, 2020.

Salimans, T. and Knowles, D. A. On using control variates with stochastic approximation for variational Bayes and its connection to stochastic linear regression. *arXiv preprint arXiv:1401.1022*, 2014.

Si, S., Oates, C., Duncan, A. B., Carin, L., and Briol, F.-X. Scalable control variates for Monte Carlo methods via stochastic optimization. *arXiv preprint arXiv:2006.07487*, 2020.

Sohl-Dickstein, J., Battaglino, P., and DeWeese, M. R. Minimum probability flow learning. In *International Conference on Machine Learning*, pp. 905–912, 2011.

South, L. F., Oates, C. J., Mira, A., and Drovandi, C. Regularised zero-variance control variates for high-dimensional variance reduction. *arXiv preprint arXiv:1811.05073*, 2018.

Stein, C. A bound for the error in the normal approximation to the distribution of a sum of dependent random variables. In *Proceedings of the Sixth Berkeley Symposium on Mathematical Statistics and Probability, Volume 2: Probability Theory*. The Regents of the University of California, 1972.

Stein, C., Diaconis, P., Holmes, S., and Reinert, G. Use of exchangeable pairs in the analysis of simulations. *Lecture Notes-Monograph Series*, pp. 1–26, 2004.

Stirzaker, D. *Stochastic processes and models*. OUP Oxford, 2005.

Titsias, M. K. and Lázaro-Gredilla, M. Doubly stochastic variational Bayes for non-conjugate inference. In *International Conference on Machine Learning*, 2014.

Titsias, M. K. and Lázaro-Gredilla, M. Local expectation gradients for black box variational inference. *Advances in neural information processing systems*, 28:2638–2646, 2015.

Titsias, M. K. and Shi, J. Double control variates for gradient estimation in discrete latent variable models. *arXiv preprint arXiv:2111.05300*, 2021.

Tokui, S. and Sato, I. Evaluating the variance of likelihood-ratio gradient estimators. In *International Conference on Machine Learning*, pp. 3414–3423, 2017.

Tucker, G., Mnih, A., Maddison, C. J., and Sohl-Dickstein, J. REBAR: Low-variance, unbiased gradient estimates for discrete latent variable models. In *Advances in Neural Information Processing Systems*, 2017.

Weaver, L. and Tao, N. The optimal reward baseline for gradient-based reinforcement learning. In *Conference on Uncertainty in Artificial Intelligence*, pp. 538–545, 2001.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Yang, J., Liu, Q., Rao, V., and Neville, J. Goodness-of-fit testing for discrete distributions via Stein discrepancy. In *International Conference on Machine Learning*, pp. 5561–5570. PMLR, 2018.

Yin, M. and Zhou, M. ARM: Augment-REINFORCE-merge gradient for stochastic binary networks. In *International Conference on Learning Representations*, 2019.

Yin, M., Ho, N., Yan, B., Qian, X., and Zhou, M. Probabilistic best subset selection via gradient-based optimization, 2020.

A. Sample-dependent Baselines in REBAR and RELAX

We start with the REINFORCE estimator with the sample-dependent baseline b_k:

$$\frac{1}{K}\sum_{k=1}^{K}(f(x_k)-b_k)\nabla_\eta \log q_\eta(x_k)+\mathbb{E}[b_k\nabla_\eta \log q_\eta(x_k)]. \tag{21}$$

REBAR (Tucker et al., 2017) introduces indirect independence on x_k in b_k through the continuous reparameterization $x = H(z), z_k \sim q_\eta(z|x = x_k)$, where z is a continuous variable and H is an argmax-like thresholding function. Specifically, $b_k = f(\sigma_\lambda(z_k))$, where σ_λ is a continuous relaxation of H controlled by the parameter λ. The correction term decomposes into two parts:

$$\begin{aligned}&\mathbb{E}_{x_k}[\mathbb{E}_{z_k|x_k}[f(\sigma_\lambda(z_k))]\nabla_\eta \log q_\eta(x_k)]\\ &= \nabla_\eta\mathbb{E}_{q_\eta(z)}[f(\sigma_\lambda(z))] - \mathbb{E}_{x_k}[\nabla_\eta\mathbb{E}_{q_\eta(z_k|x_k)}[f(\sigma_\lambda(z_k))]].\end{aligned}$$

Both parts can be estimated with the reparameterization trick (Kingma & Welling, 2014; Rezende et al., 2014; Titsias & Lázaro-Gredilla, 2014) which often has low variance. The RELAX (Grathwohl et al., 2018) estimator generalizes REBAR by noticing that $f(\sigma_\lambda(z))$ can be replaced with a free-form differentiable function $c_\phi(z)$. However, RELAX still relies on parameterizing $c_\phi(z)$ as $f(\sigma_\lambda(z)) + r_\theta(z)$ to achieve strong performance, as noted in Dong et al. (2020).

B. Proof of Proposition 1

Recall our estimator defined in (18) is

$$\frac{1}{K}\sum_{k=1}^{K}[(f(x_k)-\frac{1}{K-1}\sum_{j\neq k}(f(x_j)+(Ah_j)(x_j)))\cdot\nabla_\eta \log q_\eta(x_k)+(A\tilde{h}'_k)(x_k)]. \tag{22}$$

To show the unbiasedness, we compute its expectation under q_η as

$$\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}_{q_\eta}[f(x_k)\nabla_\eta \log q_\eta(x_k)]-\frac{1}{K(K-1)}\sum_{k=1}^{K}\sum_{j\neq k}\mathbb{E}_{q_\eta}[(f(x_j)+(Ah_j)(x_j))\nabla_\eta \log q_\eta(x_k)]+\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}_{q_\eta}[(A\tilde{h}'_k)(x_k)].$$

Since the first term is the desired gradient $\nabla_\eta\mathbb{E}_{q_\eta}[f(x)]$ and the third term is zero, it suffices to show that the second term also vanishes. Using the law of total expectations, we find for $j \neq k$,

$$\begin{aligned}\mathbb{E}_{q_\eta}[(f(x_j)+(Ah_j)(x_j))\nabla_\eta \log q_\eta(x_k)] &= \mathbb{E}_{x_k\sim q_\eta}[\mathbb{E}_{q_\eta}[f(x_j)+(Ah_j)(x_j)\mid x_k]\nabla_\eta \log q_\eta(x_k)]\\ &= \mathbb{E}_{x_k\sim q_\eta}[\mathbb{E}_{q_\eta}[f(x_j)\mid x_k]\nabla_\eta \log q_\eta(x_k)]\\ &= \mathbb{E}_{q_\eta}[f(x_j)\nabla_\eta \log q_\eta(x_k)]\\ &= \mathbb{E}_{x_j\sim q_\eta}[f(x_j)\mathbb{E}_{x_k\sim q_\eta}[\nabla_\eta \log q_\eta(x_k)\mid x_j]] = 0,\end{aligned}$$

which completes the proof.

C. Experimental Details

C.1. Details of VAE experiments

The VAE has two hidden layers with 200 units activated by LeakyReLU with the coefficient 0.3. To optimize the VAE we use Adam with base learning rate 10^{-4} for non-binarized data and 10^{-3} for dynamically binarized data, except for binarized Fashion-MNIST we decreased the learning rate to 3×10^{-4} because otherwise the training is very unstable for all estimators. We use Adam with the same learning rate 10^{-3} for adapting our control variate network in all experiments. The batch size is 100. The settings of other estimators are kept the same with Titsias & Shi (2021).

For the minimum probability flow (MPF) Stein estimator (9) and the difference Stein estimator (12) illustrated in Figure 3, we replace the term $\frac{q(y)}{q(x)}$ with $\frac{q(y)}{q(x)+10^{-3}}$ to alleviate numerical instability. In the MPF Stein operator, we choose the neighborhood $\mathcal{N}_x$ to be the states that differ in only one coordinate from x. In our experiments, we find that the difference Stein estimator is highly unstable and may diverge as the iteration proceeds.

C.2. Details of hierarchical VAE experiments

We optimize the hierarchical VAE using Adam with base learning rate 10^{-4}. Our control variate network is optimized using Adam with learning rate 10^{-3}. Settings of training multilayer VAEs are kept the same with Dong et al. (2020), except that we do not optimize the prior distribution of the VAE hidden layer, and use 100 samples in each step.

D. Additional Results

Table 2. Average test 100-sample bounds of binary latent VAEs trained with $K = 2, 3, 4$ (except for RELAX which uses 3 evaluations) on MNIST, Fashion-MNIST, and Omniglot. We report the average value ± 1 standard error after 1M steps over 5 independent runs.

		Bernoulli Likelihoods			Gaussian Likelihoods		
		MNIST	Fashion-MNIST	Omniglot	MNIST	Fashion-MNIST	Omniglot
$K=2$	DisARM	-101.61 ± 0.07	-239.11 ± 0.11	-118.34 ± 0.05	669.26 ± 0.53	163.40 ± 0.59	305.32 ± 0.89
	Double CV	-100.91 ± 0.04	-239.00 ± 0.17	-118.45 ± 0.09	677.02 ± 0.93	164.99 ± 0.71	304.72 ± 1.39
	RODEO (Ours)	$\mathbf{-100.78 \pm 0.16}$	$\mathbf{-238.97 \pm 0.09}$	$\mathbf{-118.09 \pm 0.05}$	$\mathbf{681.11 \pm 0.31}$	$\mathbf{168.26 \pm 0.73}$	$\mathbf{308.55 \pm 1.02}$
$K=3$	ARMS	-99.08 ± 0.12	-238.19 ± 0.11	-116.78 ± 0.13	688.61 ± 0.84	174.14 ± 0.44	320.45 ± 1.07
	Double CV	-99.16 ± 0.12	-238.54 ± 0.16	-116.75 ± 0.15	690.28 ± 0.49	173.67 ± 0.30	322.88 ± 1.10
	RODEO (Ours)	$\mathbf{-98.72 \pm 0.14}$	$\mathbf{-237.97 \pm 0.12}$	$\mathbf{-116.69 \pm 0.09}$	$\mathbf{695.11 \pm 0.33}$	$\mathbf{174.57 \pm 0.30}$	$\mathbf{323.92 \pm 1.24}$
RELAX (3 evals)		-100.80 ± 0.09	-239.03 ± 0.11	-117.60 ± 0.06	686.21 ± 0.57	171.43 ± 0.61	317.78 ± 1.25
$K=4$	ARMS	-98.10 ± 0.11	-237.78 ± 0.13	-116.04 ± 0.04	695.74 ± 0.42	178.70 ± 0.83	$\mathbf{330.65 \pm 1.07}$
	Double CV	-97.94 ± 0.19	-237.78 ± 0.14	-116.13 ± 0.14	697.32 ± 1.03	179.19 ± 0.59	329.37 ± 1.52
	RODEO (Ours)	$\mathbf{-97.56 \pm 0.13}$	$\mathbf{-237.66 \pm 0.15}$	$\mathbf{-116.03 \pm 0.08}$	$\mathbf{699.19 \pm 0.40}$	$\mathbf{179.32 \pm 0.54}$	329.14 ± 2.06

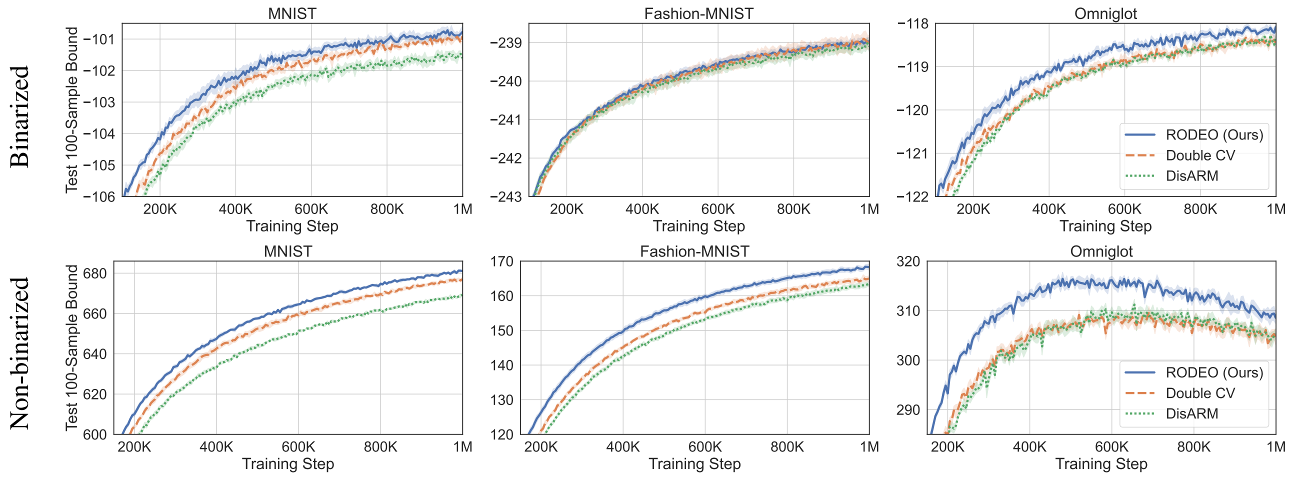

Figure 6. Average 100-sample bounds on test data for binary latent VAEs trained on (top) dynamically binarized and (bottom) non-binarized MNIST, Fashion-MNIST, and Omniglot using $K = 2$.

Table 3. Average running time across 1000 iterations on an NVIDIA 3080Ti GPU for the VAE experiments in Section 6.1.

	Double CV	DisARM/ARMS	RODEO (Ours)	RELAX (3 evals)
$K=2$	2.2 ms/step	2.2 ms/step	2.8 ms/step	4.1 ms/step
$K=3$	2.4 ms/step	2.3 ms/step	4.8 ms/step	
$K=4$	2.4 ms/step	2.3 ms/step	7.4 ms/step	

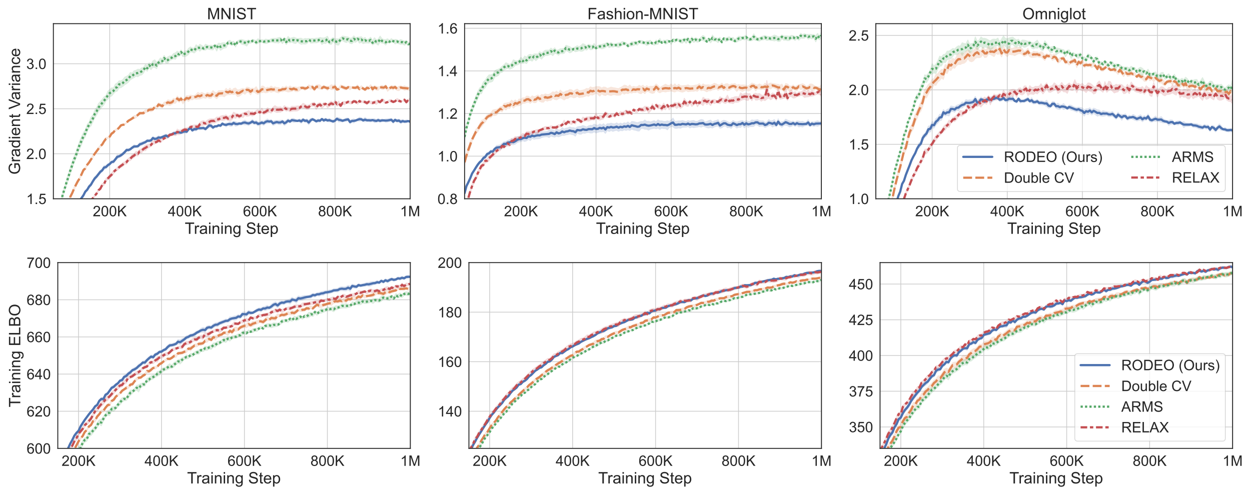

Figure 7. Training binary latent VAEs with Gaussian likelihoods with three evaluations of f per step using RODEO/Double CV/ARMS with $K = 3$ or RELAX on non-binarized MNIST, Fashion-MNIST, and Omniglot. (Top) variance of gradient estimates. (Bottom) the plot of average ELBO on training examples against training steps.

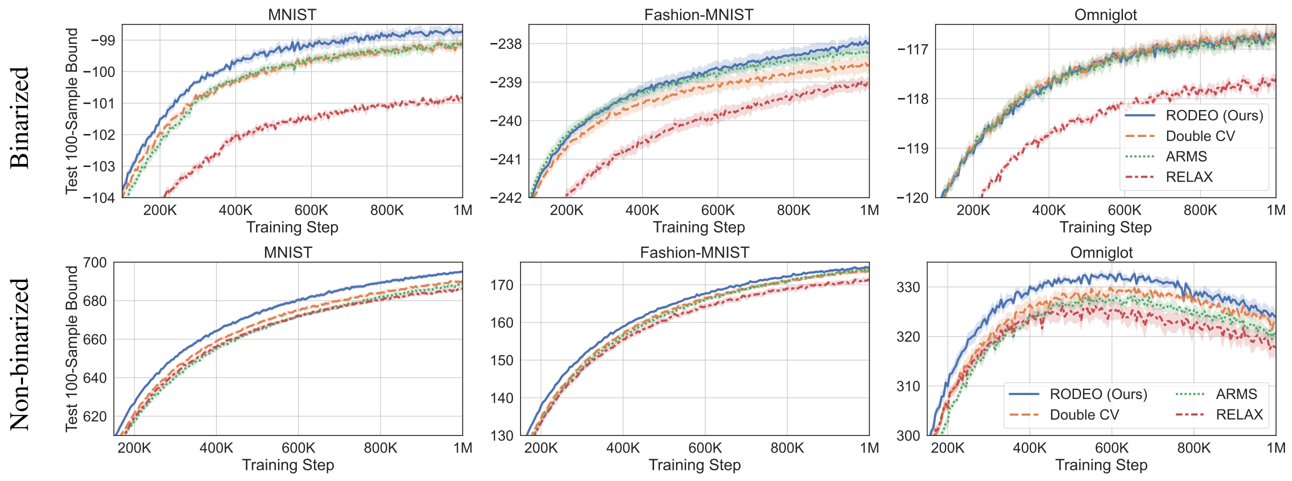

Figure 8. Average 100-sample bounds on test data for binary latent VAEs trained on (top) dynamically binarized and (bottom) non-binarized MNIST, Fashion-MNIST, and Omniglot with three evaluations of f per step using RODEO/Double CV/ARMS with $K = 3$ or RELAX.

Table 4. Average running time across 1000 iterations on an NVIDIA 3080Ti GPU when training hierarchical VAEs with $K = 2$.

	Double CV	DisARM/ARMS	RODEO (Ours)
Two layers	3.4 ms/step	2.7 ms/step	5.8 ms/step
Three layers	5.3 ms/step	4.5 ms/step	9.3 ms/step
Four layers	7.9 ms/step	6.4 ms/step	13.1 ms/step

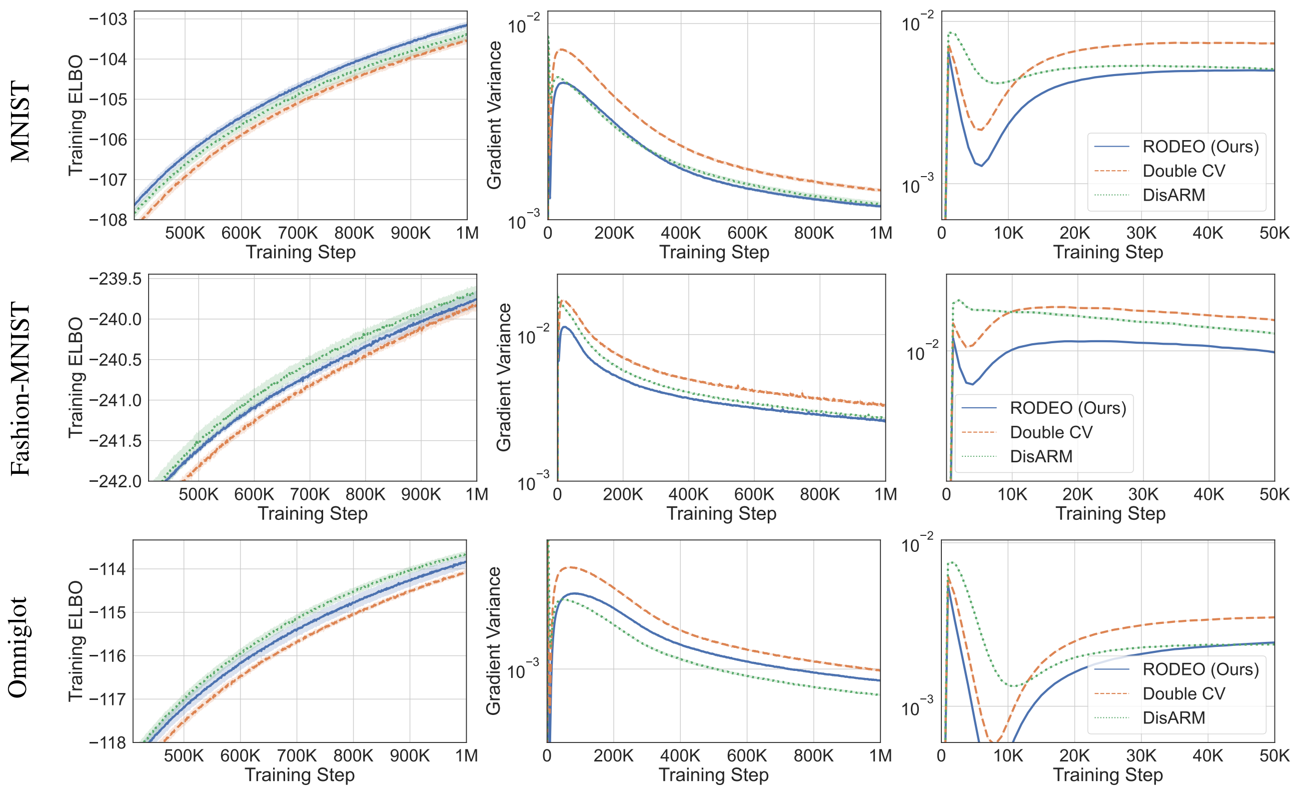

Figure 9. Training hierarchical binary latent VAEs with **two** stochastic layers on MNIST, Fashion-MNIST and Omniglot (from top to bottom). We plot (left) the average ELBO on training examples and (middle) variance of gradient estimates. We zoom into the first 50K steps of the variance plot on the right figure.

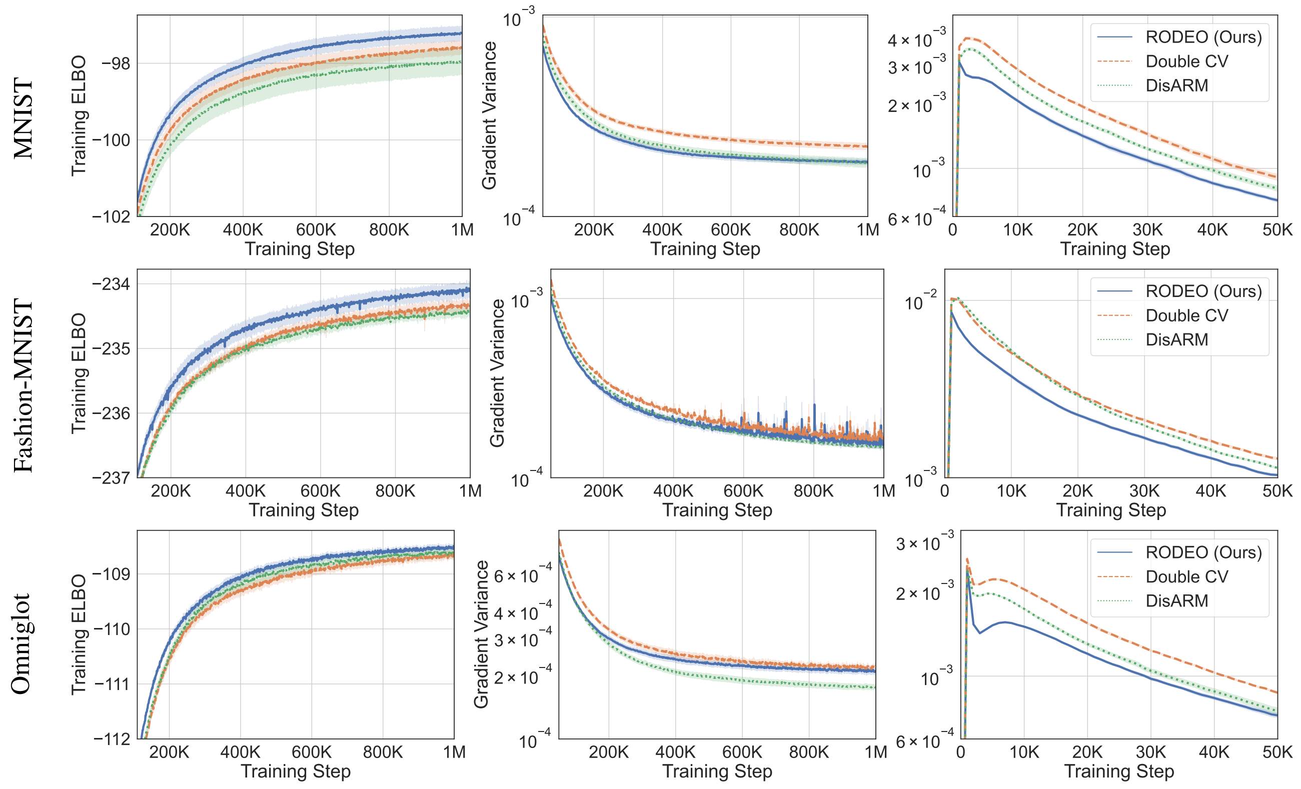

Figure 10. Training hierarchical binary latent VAEs with **three** stochastic layers. We plot the average ELBO on training examples (left) and variance of gradient estimates (middle). We zoom into the first 50K steps of the variance plot on the right figure.

Table 5. Training hierarchical binary latent VAEs on dynamically binarized MNIST, Fashion-MNIST, and Omniglot. We report the average ELBOs (± 1 standard error) on the training set and 100-sample bounds on test data after 1M steps over 5 independent runs.

		Training ELBO			Test Log-Likelihood		
		MNIST	Fashion-MNIST	Omniglot	MNIST	Fashion-MNIST	Omniglot
Two layers	Double CV	-103.52 ± 0.06	-239.82 ± 0.07	-114.06 ± 0.04	-97.62 ± 0.08	-237.65 ± 0.06	-110.48 ± 0.04
	DisARM	-103.39 ± 0.12	$\mathbf{-239.67 \pm 0.06}$	$\mathbf{-113.67 \pm 0.05}$	-97.56 ± 0.07	$\mathbf{-237.61 \pm 0.06}$	$\mathbf{-110.15 \pm 0.04}$
	RODEO (Ours)	$\mathbf{-103.15 \pm 0.07}$	-239.76 ± 0.09	-113.84 ± 0.11	$\mathbf{-97.43 \pm 0.03}$	-237.63 ± 0.07	-110.32 ± 0.10
Three layers	Double CV	-97.59 ± 0.15	-234.34 ± 0.07	-108.66 ± 0.06	-93.71 ± 0.12	-234.34 ± 0.07	-107.48 ± 0.07
	DisARM	-97.95 ± 0.30	-234.45 ± 0.05	-108.60 ± 0.08	-94.12 ± 0.28	-234.46 ± 0.06	-107.32 ± 0.10
	RODEO (Ours)	$\mathbf{-97.21 \pm 0.17}$	$\mathbf{-234.11 \pm 0.10}$	$\mathbf{-108.51 \pm 0.04}$	$\mathbf{-93.52 \pm 0.16}$	$\mathbf{-234.19 \pm 0.07}$	$\mathbf{-107.26 \pm 0.06}$
Four layers	Double CV	-98.73 ± 0.06	-235.69 ± 0.07	-110.92 ± 0.06	-93.28 ± 0.03	-234.63 ± 0.03	-107.86 ± 0.03
	DisARM	-98.97 ± 0.02	-235.50 ± 0.04	-110.85 ± 0.07	-93.56 ± 0.04	-234.52 ± 0.04	-107.87 ± 0.05
	RODEO (Ours)	$\mathbf{-98.67 \pm 0.14}$	$\mathbf{-235.39 \pm 0.05}$	$\mathbf{-110.79 \pm 0.03}$	$\mathbf{-93.27 \pm 0.09}$	$\mathbf{-234.39 \pm 0.06}$	$\mathbf{-107.77 \pm 0.02}$