
Rao-Blackwellised Reparameterisation Gradients

Kevin H. Lam
Department of Statistics
University of Oxford

Thang D. Bui
School of Computing
Australian National University

George Deligiannidis
Department of Statistics
University of Oxford

Yee Whye Teh
Department of Statistics
University of Oxford

{lam,deligian,y.w.teh}@stats.ox.ac.uk, thang.bui@anu.edu.au

Abstract

Latent Gaussian variables have been popularised in probabilistic machine learning. In turn, gradient estimators are the machinery that facilitates gradient-based optimisation for models with latent Gaussian variables. The reparameterisation trick is often used as the default estimator as it is simple to implement and yields low-variance gradients for variational inference. In this work, we propose the R2-G2 estimator as the Rao-Blackwellisation of the reparameterisation gradient estimator. Interestingly, we show that the local reparameterisation gradient estimator for Bayesian MLPs is an instance of the R2-G2 estimator and Rao-Blackwellisation. This lets us extend benefits of Rao-Blackwellised gradients to a suite of probabilistic models. We show that initial training with R2-G2 consistently yields better performance in models with multiple applications of the reparameterisation trick.

1 Introduction

Latent random variables are ubiquitous in probabilistic machine learning (ML) as they enable us to embed prior assumptions into models, such as uncertainty in model parameters and low-dimensional latent representations of observed data structures. Thus, latent random variables appear in a range of modelling tasks, including variational inference, generative modelling, and density estimation. A wide variety of modern models with latent random variables, particularly those that parameterise a neural network (NN), are trained with gradient-based optimisation. This presents a non-trivial problem on how analytical derivatives can incorporate stochasticity, and has led to substantial interest in *gradient estimators*. These estimators enable gradient-based optimisation by computing approximations of gradients that are compatible with automatic differentiation software. For comprehensive reviews of gradient estimators with latent continuous and discrete random variables, we refer the reader to Mohamed et al. (2020) and Huijben et al. (2023) respectively.

We revisit the class of gradient estimators for latent Gaussian variables (Kingma and Welling, 2014; Rezende et al., 2014; Titsias and Lázaro-Gredilla, 2014), as Gaussian variables often serve as the default distribution for the noise or prior in probabilistic machine learning tasks. These estimators, known as pathwise estimators or the reparameterisation trick, have been popularised as they produce low-variance gradients for variational inference. In particular, it has been shown in Xu et al. (2019) that the reparameterisation gradient estimator has lower variance than a Rao-Blackwellised REINFORCE estimator (Williams, 1992) for variational inference. These estimators are also attractive as they are *single-sample* estimators since training models with them only requires sampling operations be done

once. In turn, this also limits costly function evaluations needed for training as models grow larger in the era of deep learning.

In this work, we present the R2-G2 estimator as an extension of the reparameterisation gradient estimator formed by conditioning on pre-activations in backpropagation. Notably, we show that the local reparameterisation gradient estimator proposed in Kingma et al. (2015) is an instance of the R2-G2 estimator for linear layers in a Bayesian MLP with independent weights and establish that it is equivalent to a Rao-Blackwellised reparameterisation gradient estimator.

Our main contributions are as follows:

- We present R2-G2, as a novel, general-purpose and single-sample gradient estimator for latent Gaussian variables as the Rao-Blackwellised reparameterisation gradient estimator;
- We show that the local reparameterisation gradient estimator is an instance of the R2-G2 estimator and enjoys variance reduction in gradients from Rao-Blackwellisation;
- We empirically show that initial training with the R2-G2 estimator consistently yields higher likelihoods on Bayesian Neural Networks and higher ELBOs on hierarchical Variational Autoencoders than the reparameterisation gradient estimator.

2 Problem setting

Let $\mathbf{v} \in \mathbb{R}^n$ be a vector of continuous random variables where we have $\mathbf{v} \sim q_{\boldsymbol{\theta}}$ for some probability distribution q parameterised by a vector $\boldsymbol{\theta}$. Suppose we are provided a continuously differentiable loss function $\ell : \mathbb{R}^n \rightarrow \mathbb{R}$ that depends on the random variables \mathbf{v} . To enable gradient-based training of a parametric model f , our goal is to compute the gradient of the expected loss

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\theta}}} [\ell_{\mathcal{D},f}(\mathbf{v})] \quad (1)$$

where \mathcal{D} is a dataset and f is a parametric function optimised by the loss such as a neural network (NN), with both affecting the evaluation of ℓ (i.e. $\ell_{\mathcal{D},f}$). We note that \mathcal{D} is described generally here to encompass both supervised and unsupervised problems. For ease of notation in the remainder of this work, we will shorten our notation of the loss from $\ell_{\mathcal{D},f}(\mathbf{v}, \boldsymbol{\theta})$ to ℓ where the context is clear with its dependence on \mathbf{v} , $\boldsymbol{\theta}$, \mathcal{D} , and f implied. Popular choices of ℓ aim to maximise the log-likelihood or the evidence lower bound (ELBO) for the dataset \mathcal{D} . The former involves setting ℓ as the negative log-likelihood ℓ_{NLL} . The latter sets ℓ as

$$\ell_{ELBO}(\mathbf{v}) = \ell_{NLL}(\mathcal{D}|\mathbf{v}) + \log \left(\frac{q_{\boldsymbol{\theta}}(\mathbf{v}|\mathcal{D})}{p(\mathbf{v})} \right)$$

where p is a prior distribution of \mathbf{v} and setting $q_{\boldsymbol{\theta}}$ as the posterior of \mathbf{v} or its approximation, making Equation 1 equivalent to maximising the ELBO. Moreover, it is equivalent to variational Bayesian inference when the form of $q_{\boldsymbol{\theta}}$ is restricted as it becomes an approximation of the posterior.

In practice, we use a Monte Carlo approximation $\mathbb{E}_{q_{\boldsymbol{\theta}}} [\ell(\mathbf{v}, \boldsymbol{\theta})] \approx \frac{1}{M} \sum_{i=1}^M \ell(\mathbf{v}^{(i)}, \boldsymbol{\theta})$ to write

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\theta}}} [\ell(\mathbf{v})] \approx \frac{1}{M} \sum_{i=1}^M \nabla_{\boldsymbol{\theta}} \ell(\mathbf{v}^{(i)})$$

where $\mathbf{v}^{(1)}, \dots, \mathbf{v}^{(M)} \sim q_{\boldsymbol{\theta}}$. However, this would require M evaluations of the loss ℓ which is not desirable when the underlying model f is large or costly to evaluate. In this work, we focus on the setting where \mathbf{v} are *independent* Gaussian random variables with the aim to derive a *single-sample* gradient estimator (i.e. $M = 1$) that is unbiased and enjoys reduced variance.

3 Related work

In this section, we revisit the single-sample gradient estimators compatible with Gaussian random variables, namely REINFORCE, reparameterisation trick and local reparameterisation trick.

REINFORCE The REINFORCE gradient estimator, also known as the *score function estimator*, is proposed by using the log-derivative trick: $\nabla_{\theta} \log q_{\theta}(\mathbf{v}) = \frac{\nabla_{\theta} q_{\theta}(\mathbf{v})}{q_{\theta}(\mathbf{v})}$ which comes as a result of the chain rule (Glynn, 1990; Williams, 1992). Formally, the partial derivatives of parameters θ are given by the REINFORCE estimator

$$\widehat{\nabla_{\theta} \ell}^{SCORE} = \ell \cdot (\nabla_{\theta} \log q_{\theta})$$

It is an unbiased estimator of $\nabla_{\theta} \mathbb{E}_{q_{\theta}}[\ell]$, and only requires that one is able to evaluate q_{θ} and sample from it. The latter requirement is not restrictive and easily achieved in most settings since the form of q_{θ} is often assumed as part of model specification such as variational inference. Despite these desirable properties, it is well-known that the REINFORCE estimator suffers from high variance (Greensmith et al., 2001; Xu et al., 2019).

Reparameterisation Trick The reparameterisation trick is compatible with random variables that have a location-scale parameterisation or have tractable inverse cumulative distribution functions (CDFs) (Price, 1958; Glasserman, 2003; Kingma and Welling, 2014; Rezende et al., 2014; Titsias and Lázaro-Gredilla, 2014; Figurnov et al., 2018; Jankowiak and Obermeyer, 2018). For Gaussian random variables, they are reparameterisable using a location-scale transformation which lets us write

$$\mathbf{v} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \stackrel{d}{=} g(\boldsymbol{\epsilon}, \boldsymbol{\mu}, \mathbf{V}) = \boldsymbol{\mu} + \mathbf{V}\boldsymbol{\epsilon}$$

where $\mathbf{V} \in \mathbb{R}^{n \times n}$, $\boldsymbol{\Sigma} = \mathbf{V}\mathbf{V}^{\top}$ and $\boldsymbol{\epsilon} \sim q_0(\boldsymbol{\epsilon}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$. The latter expression for \mathbf{v} is compatible with automatic differentiation and enables gradient-based updates to $\boldsymbol{\mu}$ and \mathbf{V} . Typically, \mathbf{V} is a diagonal matrix parameterised by a vector of variances $\boldsymbol{\tau} = (\sigma_1^2, \dots, \sigma_n^2)$. That is, $\mathbf{V} = \boldsymbol{\Sigma}^{\frac{1}{2}}$ with $\boldsymbol{\Sigma} = \text{diag}(\boldsymbol{\tau})$, so we have $\theta = \{\boldsymbol{\mu}, \boldsymbol{\tau}\}$ as our training parameters. Formally, we can write $\ell(g(\boldsymbol{\epsilon}, \theta))$ and apply the chain rule to yield the reparameterisation gradient estimator

$$\widehat{\nabla_{\theta} \ell}^{RT} = (J_{\ell}(\mathbf{v}) \cdot [\mathbf{I}_n \mid \frac{1}{2}\boldsymbol{\Sigma}^{-\frac{1}{2}} \odot (\mathbf{1}_n \boldsymbol{\epsilon}^{\top})])^{\top} \quad (2)$$

where J_{ℓ} is the Jacobian of ℓ with respect to \mathbf{v} , and $[A|B]$ denotes a partitioned matrix with block matrices A and B . The terms in the matrix are derived from the Jacobians $J_g(\boldsymbol{\mu}) = \mathbf{I}_n$ and $J_g(\boldsymbol{\tau}) = \frac{1}{2}\boldsymbol{\Sigma}^{-\frac{1}{2}} \odot (\mathbf{1}_n \boldsymbol{\epsilon}^{\top})$. Equation 2 is an unbiased estimator of $\nabla_{\theta} \mathbb{E}_{q_{\theta}}[\ell]$ due to the equivalence in distribution achieved by the reparameterisation trick. In the context of probabilistic modelling, the reparameterisation trick is often applied to individual scalar inputs of a decoder within a variational autoencoder (VAE) or individual weights in a Bayesian neural network (BNN). The latter may also be referred to as the *global* reparameterisation trick.

Local Reparameterisation Trick The local reparameterisation trick was proposed for NNs where the *weights of linear layers* are independent Gaussian random variables (Kingma et al., 2015). Given an input $\mathbf{x} \in \mathbb{R}^n$, the pre-activations of these linear layers are given by

$$\mathbf{z} = \begin{bmatrix} \mathbf{x}^{\top} g(\boldsymbol{\epsilon}^{(1)}, \boldsymbol{\theta}^{(1)}) \\ \vdots \\ \mathbf{x}^{\top} g(\boldsymbol{\epsilon}^{(m)}, \boldsymbol{\theta}^{(m)}) \end{bmatrix} \in \mathbb{R}^m$$

where $\boldsymbol{\epsilon}^{(i)}$ and $\boldsymbol{\theta}^{(j)}$ are independent for all $i \neq j$, and they admit a factorised Gaussian distribution $\tilde{q}_{\mathbf{z}} = \prod_{i=1}^m \tilde{q}_{z_i}$ where each pre-activation $z_i \sim \tilde{q}_{z_i}$. Instead of sampling g and computing $\mathbf{x}^{\top} g$, it is more efficient to *locally* apply the reparameterisation trick to directly sample scalar pre-activations

$$z_i \sim \mathcal{N}\left(\sum_{j=1}^n x_j \mu_j^{(i)}, \sum_{j=1}^n x_j^2 (\sigma_j^{(i)})^2\right) \stackrel{d}{=} \sum_{j=1}^n x_j \mu_j^{(i)} + \left(\sum_{j=1}^n x_j^2 (\sigma_j^{(i)})^2\right)^{\frac{1}{2}} \xi_i \quad (3)$$

where $\boldsymbol{\mu}^{(i)}$ and $\boldsymbol{\tau}^{(i)} = ((\sigma_1^{(i)})^2, \dots, (\sigma_n^{(i)})^2)$ are the *global* mean and variance parameters of the Gaussian variables used to compute z_i respectively, $\boldsymbol{\theta}^{(i)} = \{\boldsymbol{\mu}^{(i)}, \boldsymbol{\tau}^{(i)}\}$, and $\xi_i \sim q_0^{(i)} = \mathcal{N}(0, 1)$. The latter expression in Equation 3 is known as the local reparameterisation trick as it changes the parameterisation of each z_i as a scalar function of $\boldsymbol{\epsilon}^{(i)}$ and $\boldsymbol{\theta}^{(i)}$. Formally, we can write $\ell(g(\boldsymbol{\epsilon}, \theta)) = \tilde{\ell}(z_1(\xi_1, \boldsymbol{\theta}^{(1)}), \dots, z_m(\xi_m, \boldsymbol{\theta}^{(m)}))$ where $\tilde{\ell} : \mathbb{R}^m \rightarrow \mathbb{R}$ denotes functions applied to \mathbf{z} to compute

Table 1: Examples of linear maps applied to Gaussian variables in probabilistic neural networks.

Application	\mathbf{W}	$\mathbf{V}\epsilon$
BNN	Outputs of a previous hidden layer	Gaussian weights in a linear layer
BNN	Vectorised patches of an image	Gaussian weights in a convolutional layer
VAE	Linear layer after reparameterisation	Gaussian latent variables

the loss ℓ (i.e. non-linearities and upper NN layers), and apply the chain rule to yield the local reparameterisation gradient estimator for each $\theta^{(i)}$

$$\widehat{\nabla_{\theta^{(i)}} \ell}^{LRT} = \left(\frac{\partial \tilde{\ell}}{\partial z_i} \cdot \left[\mathbf{x}^\top \mid \frac{1}{2} \left(\sum_{j=1}^n x_j^2 (\sigma_j^{(i)})^2 \right)^{-\frac{1}{2}} \xi_i (\mathbf{x} \odot \mathbf{x})^\top \right] \right)^\top \quad (4)$$

where \odot denotes element-wise product of matrices. The terms in the matrix are derived from the Jacobians $J_{z_i}(\mu^{(i)}) = \mathbf{x}^\top$ and $J_{z_i}(\tau^{(i)}) = \frac{1}{2} \left(\sum_{j=1}^n x_j^2 (\sigma_j^{(i)})^2 \right)^{-\frac{1}{2}} \xi_i (\mathbf{x} \odot \mathbf{x})^\top$. Equation 4 is an unbiased estimator of $\nabla_{\theta} \mathbb{E}_{q_{\theta^{(i)}}}[\ell]$ due to the equivalence in distribution achieved by the local reparameterisation trick. A notable application of the local reparameterisation trick is when performing mean-field variational inference within the linear layer of a BNN. It has been empirically observed that Equation 4 has lower variance than Equation 2 (Kingma et al., 2015). In Theorem 4.3, we show that the local reparameterisation gradient estimator is equivalent to applying Rao-Blackwellisation to the reparameterisation gradient estimator and thereby enjoys variance reduction benefits of Rao-Blackwellisation.

4 R2-G2 Gradient Estimator

The reparameterisation trick can be seen as a procedure that couples the specification of a probabilistic model with an expression for gradients. In turn, assumptions on the former would restrict the applicability of the latter. This highlights the disadvantage of the local reparameterisation trick: the probabilistic model and gradients induced by the local reparameterisation trick is not suitable for general settings where the pre-activations \mathbf{z} do not admit a factorised Gaussian distribution (i.e. the covariance matrix of \mathbf{z} is not diagonal). On the other hand, the local reparameterisation trick has been empirically shown to enjoy lower variance of gradients than the global reparameterisation trick.

To get the best of both worlds, we seek a reparameterisation gradient estimator that is general-purpose and enjoys reduced variance. To this end, we present our contribution: the **Rao-Blackwellised Reparameterisation Gradient Estimator for Gaussian random variables**, coined the R2-G2 estimator, as the Rao-Blackwellisation of the reparameterisation gradient estimator by conditioning on the realisation of multivariate Gaussian vectors resultant from linear transformations of Gaussian vectors. We first describe the Rao-Blackwellisation of the reparameterisation gradient estimator. We then provide the analytical form of the R2-G2 estimator and a summary of key properties of the R2-G2 estimator and its connection to existing methods. In particular, we show that the local reparameterisation gradient estimator is an instance of the R2-G2 estimator. We then conclude with a practical implementation to bypass costly matrix inversion operations by reformulating matrix-vector products as the solution to a quadratic optimisation problem. We defer all proofs to Appendix B.

4.1 Rao-Blackwellisation of the Reparameterisation Gradient Estimator

The idea behind our Rao-Blackwellisation scheme is to condition on linear transformations of Gaussian variables since conditional Gaussian distributions can be described analytically. This is further motivated by the observation that we can decompose loss evaluations in NNs as

$$\ell(g(\epsilon, \theta)) = (\tilde{\ell} \circ \mathbf{W})(g(\epsilon, \theta)) \quad (5)$$

where $\mathbf{W} : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is a linear map and $\mathbf{W} \cdot g(\epsilon, \theta)$ are pre-activations. This generalises the idea of the local reparameterisation trick where \mathbf{W} is a row vector (i.e. $\mathbf{W} = \mathbf{x}^\top$) and inputs of $\tilde{\ell}$ are m scalar pre-activations. In deep learning, \mathbf{W} frequently appears as hidden layers within NNs. A list of

common linear transformations in probabilistic NNs is given in Table 1. Applying the chain rule to Equation 5 gives an alternative expression of the reparameterisation gradient estimator

$$\widehat{\nabla_{\boldsymbol{\theta}} \ell}^{RT} = (J_{\tilde{\ell}}(\mathbf{W} \cdot g) \cdot \mathbf{W} \cdot [\mathbf{I}_n \mid \frac{1}{2} \boldsymbol{\Sigma}^{-\frac{1}{2}} \odot (\mathbf{1}_n \boldsymbol{\epsilon}^\top)])^\top \quad (6)$$

where $J_{\tilde{\ell}}$ is the Jacobian of $\tilde{\ell}$. With Equation 6 in hand, we now present the R2-G2 estimator¹.

Definition 4.1 (R2-G2) *The R2-G2 gradient estimator is given by*

$$\widehat{\nabla_{\boldsymbol{\theta}} \ell}^{R2-G2} = \mathbb{E}_{\tilde{q}_0} \left[\widehat{\nabla_{\boldsymbol{\theta}} \ell}^{RT}(\boldsymbol{\epsilon}) \right] = \left(J_{\tilde{\ell}}(\mathbf{W} \cdot g) \cdot \mathbf{W} \cdot \mathbb{E}_{\tilde{q}_0} [\mathbf{I}_n \mid \frac{1}{2} \boldsymbol{\Sigma}^{-\frac{1}{2}} \odot (\mathbf{1}_n \boldsymbol{\epsilon}^\top)] \right)^\top \quad (7)$$

where $\boldsymbol{\epsilon} | \mathbf{W} \cdot g = \mathbf{z} \sim \tilde{q}_0$ with mean $\boldsymbol{\epsilon}^* = \mathbf{A}^\top (\mathbf{A} \mathbf{A}^\top)^\dagger (\mathbf{z} - \mathbf{W} \boldsymbol{\mu}) = \mathbf{A}^\top (\mathbf{A} \mathbf{A}^\top)^\dagger \mathbf{A} \boldsymbol{\epsilon}$ with $\mathbf{A} = \mathbf{W} \mathbf{V}$. By linearity of expectations, it also admits a closed-form expression

$$\widehat{\nabla_{\boldsymbol{\theta}} \ell}^{R2-G2} = \left(J_{\tilde{\ell}}(\mathbf{W} \cdot g) \cdot \mathbf{W} \cdot [\mathbf{I}_n \mid \frac{1}{2} \boldsymbol{\Sigma}^{-\frac{1}{2}} \odot (\mathbf{1}_n (\boldsymbol{\epsilon}^*)^\top)] \right)^\top. \quad (8)$$

The R2-G2 estimator is a single-sample gradient estimator as it is a conditional expectation of the reparameterisation gradient estimator formed by conditioning on a single sample of the vector of pre-activations \mathbf{z} , making it the Rao-Blackwellisation (Blackwell, 1947; Rao et al., 1992) of the reparameterisation gradient estimator from Equation 6. The law of total variance lets us deduce that the R2-G2 estimator has lower variance since it swaps the random matrix in Equation 6 with its conditional expectation. We conclude our description of the R2-G2 estimator with its key properties, namely unbiasedness and enjoying lower variance than the reparameterisation gradient estimator.

Proposition 4.2 *Denote $\mathbf{z} \sim q_{\mathbf{z}} = \mathcal{N}(\mathbf{W} \cdot \boldsymbol{\mu}, \mathbf{A} \mathbf{A}^\top)$. Then we have*

$$\mathbb{E}_{q_{\mathbf{z}}} [\widehat{\nabla_{\boldsymbol{\theta}} \ell}^{R2-G2}] = \nabla_{\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\theta}}} [\ell]$$

and

$$\mathbb{E}_{q_{\mathbf{z}}} \left[\left\| \widehat{\nabla_{\boldsymbol{\theta}} \ell}^{R2-G2} - \nabla_{\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\theta}}} [\ell] \right\|^2 \right] \leq \mathbb{E}_{q_0} \left[\left\| \widehat{\nabla_{\boldsymbol{\theta}} \ell}^{RT} - \nabla_{\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\theta}}} [\ell] \right\|^2 \right].$$

4.2 Computation of Rao-Blackwellisation scheme as a least-squares problem

The R2-G2 estimator exploits the analytical form of the conditional Gaussian distribution. This is done by either sampling from the conditional Gaussian distribution \tilde{q}_0 within Equation 7 or directly using the mean of the conditional Gaussian distribution $\boldsymbol{\epsilon}^*$ in place of $\boldsymbol{\epsilon}$ within Equation 8. Equation 7 requires computing the Cholesky factor of the covariance matrix of \tilde{q}_0 , which incurs a $\mathcal{O}(m^3)$ computational cost and a $\mathcal{O}(m^2)$ storage cost. The latter makes it impractical for training NNs as each gradient descent step would require instantiating the matrix $\mathbf{A} \mathbf{A}^\top$. This makes Equation 8 more desirable as long as $\boldsymbol{\epsilon}^*$ is computed and stored efficiently. To do so, we consider the linear system

$$\mathbf{A} \mathbf{A}^\top \boldsymbol{\beta} = \mathbf{A} \boldsymbol{\epsilon}. \quad (9)$$

It can be verified that any solution $\boldsymbol{\beta}^*$ to Equation 9 satisfies $\boldsymbol{\epsilon}^* = \mathbf{A}^\top \boldsymbol{\beta}^*$ (see Appendix D).

Least-squares characterisation of Rao-Blackwellisation Observe that Equation 9 is a normal equation that describes the first-order optimality condition of the quadratic optimisation problem

$$\underset{\boldsymbol{\beta} \in \mathbb{R}^m}{\text{minimise}} \quad \frac{1}{2} \|\boldsymbol{\epsilon} - \mathbf{A}^\top \boldsymbol{\beta}\|_2^2 \Leftrightarrow \underset{\boldsymbol{\beta} \in \mathbb{R}^m}{\text{maximise}} \quad \exp \left(-\frac{1}{2} \|\boldsymbol{\epsilon} - \mathbf{A}^\top \boldsymbol{\beta}\|_2^2 \right).$$

We can interpret computing a solution $\boldsymbol{\beta}^*$ as fitting a single-sample multivariate linear model $\boldsymbol{\epsilon} | \mathbf{A}$

$$\boldsymbol{\epsilon} = \mathbf{A}^\top \boldsymbol{\beta} + \boldsymbol{\delta}$$

¹Equations 7 and 8 in Definition 4.1 only hold when \mathbf{v} are independent Gaussian variables. A more general expression can be derived when independence does not hold.

where $\beta \in \mathbb{R}^m$ and $\delta \sim \mathcal{N}(\mathbf{0}_n, \mathbf{I}_n)$. Under this linear model, the goal is to solve for β^* by using covariance parameters \mathbf{A} as *predictors* of the noise ϵ . With β^* computed, we can then compute

$$\epsilon^* = \mathbf{A}^\top \beta^* = \mathbf{A}^\top (\mathbf{A}\mathbf{A}^\top)^\dagger \mathbf{A}\epsilon \quad (10)$$

which corresponds to the expression of the mean of a conditional Gaussian distribution (see Lemma A.1). In other words, computing the R2-G2 estimator is inherently solving a least squares problem and applying a linear transformation. Intuitively, the former is the mechanism giving variance reduction. In the language of linear models, the observed noise ϵ is projected to the *fitted* noise ϵ^* which has minimal Euclidean norm.

Iterative solver By defining functions that compute matrix-vector products with the matrix $\mathbf{A}\mathbf{A}^\top$, we can use the conjugate gradient algorithm to solve Equation 9 for β^* . The conjugate gradient algorithm is an iterative method that terminates in at most $\text{rank}(\mathbf{A}) \leq m$ iterations (Kaasschieter, 1988; Nocedal and Wright, 1999; Hayami, 2018), and only requires storing the updated solution at each iteration. While the former implies that the worst-case computational costs remains at $\mathcal{O}(m^3)$, the latter implies the storage cost is reduced to $\mathcal{O}(m)$ and makes it practical to compute β^* . In our setting, the worst-case computational costs of $\mathcal{O}(m^3)$ can be reduced since the structure of \mathbf{A} is known (see Appendix F). With β^* in hand, we can compute ϵ^* and modify forward computations to use the R2-G2 gradient estimator for backpropagation², by implementing Algorithm 1 in deep learning frameworks such as PyTorch (Paszke et al., 2019).

Algorithm 1 Forward Pass with R2-G2 Gradients.

Input: matrix \mathbf{A} , noise vector ϵ .

Compute $\mathbf{z} = \mathbf{A}\epsilon$.

Compute $\beta^* = \text{conjugate_gradient}(\mathbf{A}, \mathbf{z})$.

Compute $\epsilon^* = \mathbf{A}^\top \beta^*$.

Compute $\mathbf{z}^* = \mathbf{A}\epsilon^*$.

Output: $\text{stop_gradient}(\mathbf{z} - \mathbf{z}^*) + \mathbf{z}^*$.

4.3 Connections to related work

Local Reparameterisation Gradient Estimator By setting $\mathbf{V}^{(i)} = (\text{diag}(\tau^{(i)}))^{\frac{1}{2}}$, $\mathbf{W} = \mathbf{x}^\top$ and the conditional distribution $\epsilon^{(i)}|z_i \sim \tilde{q}_0^{(i)}$ for $i = 1, \dots, m$ within Definition 4.1, we are able to show that the local reparameterisation gradient estimator is equivalent to the R2-G2 estimator. Theorem 4.3 presents this equivalence and formalises the empirical variance reduction of the local reparameterisation trick observed in the experiments of Kingma et al. (2015), as an instance of Rao-Blackwellisation and the R2-G2 estimator. Formally, it shows the local reparameterisation trick is equivalent to using pre-activation samples for forward computations and a Rao-Blackwellised *global* reparameterisation gradient estimator to update parameters in linear layers.

Theorem 4.3 Suppose we have a BNN linear layer where weights are independent Gaussian random variables. That is, $\theta^{(i)} = \{\mu^{(i)}, \tau^{(i)}\}$ where $\tau^{(i)} = \left((\sigma_1^{(i)})^2, \dots, (\sigma_n^{(i)})^2 \right)$ for $i = 1, \dots, m$.

Then for each $i = 1, \dots, m$, we have $\widehat{\nabla_{\theta^{(i)}} \ell}^{\text{R2-G2}} = \mathbb{E}_{\tilde{q}_0^{(i)}} \left[\widehat{\nabla_{\theta^{(i)}} \ell}^{\text{RT}} \right] \stackrel{d}{=} \widehat{\nabla_{\theta^{(i)}} \ell}^{\text{LRT}}$ and

$$\mathbb{E}_{\tilde{q}_{z_i}} \left[\left\| \widehat{\nabla_{\theta^{(i)}} \ell}^{\text{LRT}} - \nabla_{\theta^{(i)}} \mathbb{E}_{q_{\theta^{(i)}}}[\ell] \right\|^2 \right] \leq \mathbb{E}_{q_0^{(i)}} \left[\left\| \widehat{\nabla_{\theta^{(i)}} \ell}^{\text{RT}} - \nabla_{\theta^{(i)}} \mathbb{E}_{q_{\theta^{(i)}}}[\ell] \right\|^2 \right].$$

As an analogy, we can view the way that R2-G2 generalises the local reparameterisation gradient estimator, in the way that square matrix inversion generalises scalar inversion. Recall that Equation 3 exploits the fact that the Gaussian pre-activations \mathbf{z} of mean-field BNN linear layers have a diagonal covariance matrix *by design*. For forward computations, this means each z_i can be efficiently sampled

with the reparameterisation trick by using the square root of the variance $\sigma_{z_i}^2 = \sum_{j=1}^n x_j^2 (\sigma_j^{(i)})^2$.

The subtle benefit of reduced variance in gradients is due to the square root function having a derivative which matches its reciprocal (up to a scaling factor). This means that the variance $\sigma_{z_i}^2$ (i.e. a scalar) is *inverted* when using the local reparameterisation gradient estimator. The R2-G2 estimator naturally extends the inversion of scalar variances to square covariance matrices, by calling the conjugate gradient algorithm, thereby making it suitable for other probabilistic models.

²In Algorithm 1, computing \mathbf{z}^* ensures automatic differentiation calculates Equation 8 for backpropagation, while the output $\text{stop_gradient}(\mathbf{z} - \mathbf{z}^*) + \mathbf{z}^*$ ensures the forward computation is still performed with \mathbf{z} .

Table 2: Log-likelihoods and classification accuracies (%) of BNNs using the R2-G2, Reparameterisation (RT) and Local Reparameterisation (LRT) estimators over 5 runs. Higher is better. Error bars denote ± 1.96 standard errors ($\sigma/\sqrt{5}$) over 5 runs. See text for details.

Dataset	Estimator	Log-likelihood		Accuracy	
		Train	Test	Train	Test
MNIST	R2-G2	-3.00 ± 0.00	-3.06 ± 0.00	99.81 ± 0.03	98.00 ± 0.08
	LRT	-3.00 ± 0.00	-3.06 ± 0.00	99.81 ± 0.04	97.99 ± 0.07
	RT	-3.00 ± 0.00	-3.07 ± 0.00	99.85 ± 0.04	98.00 ± 0.11
CIFAR-10	R2-G2	-3.83 ± 0.01	-3.87 ± 0.01	70.08 ± 0.43	67.97 ± 0.66
	RT	-3.84 ± 0.01	-3.88 ± 0.02	69.76 ± 0.46	67.98 ± 0.88

Variance reduction as Stochastic Linear Regression Variance reduction through stochastic linear regression has previously been explored in the context of variational inference Salimans and Knowles (2013, 2014); Salimans (2014). Specifically, variational inference is presented as fitting linear regression with the unnormalised log posterior as the dependent variable and the sufficient statistics of latent random variables as explanatory variables. The connection to variance reduction of gradients is then made by using *multiple* samples to construct control variates that correlate with the gradient of the KL-divergence. This differs from our work as we present variance reduction of gradients as Rao-Blackwellisation through the R2-G2 estimator as a *single-sample* gradient estimator, and explicate the connection to linear regression with the reparameterisation noise and covariance matrix parameters as the dependent and explanatory variables respectively.

5 Experiments

5.1 Protocol

The R2-G2 estimator can be readily applied to any existing application of the reparameterisation trick for Gaussian variables. While this may appear restrictive, we note that Gaussian distributions often serve as a default prior or noise distribution in probabilistic ML tasks. To motivate our experiments, we note that the R2-G2 and reparameterisation gradient estimators would yield the same model when a *large* number of gradient descent steps are taken, as the mean gradient of both estimators are the same. The focus of our experiments is during initial training, where only a *small* number of gradient descent steps are taken. Examples of scenarios where this setting can be beneficial include pre-training to discover a better initialisation for a model, fine-tuning a pre-trained model, or when the amount of training is limited by a computation budget.

We evaluate the benefits of initial training with the R2-G2 estimator for probabilistic models. We consider two standard tasks with variational Bayesian models that utilise a mean-field approximation of the posterior: image classification with BNNs and generative modelling with hierarchical VAEs. In our experiments, we provide numerical comparisons against the reparameterisation (RT) and local reparameterisation (LRT) gradient estimators, where permissible. We do not compare against the REINFORCE estimator as it is well-known that its high variance makes optimisation difficult. Unless stated otherwise, we compute pre-activations in the same way as the reparameterisation trick (i.e. sampling $g(\epsilon, \theta)$ and computing $\mathbf{W} \cdot g$). We defer the full details of our experiments to Appendix E.

5.2 Image classification with Bayesian Neural Networks

We consider fully stochastic BNNs for image classification tasks on two standard benchmark datasets: MNIST (LeCun et al., 2010) and CIFAR-10 (Krizhevsky and Hinton, 2009). In this setting, the input data \mathcal{D} is a dataset and the latent variables \mathbf{v} are weights (i.e. \mathbf{v} are global variables). To enable training with minibatches, we use the stochastic approximation of the variational lower bound

$$\log p(y|\mathbf{x}) > \mathbb{E}_{q_{\theta}(\mathbf{v}|\mathcal{D})} \left[\log \left(\frac{p(\mathbf{v})}{q_{\theta}(\mathbf{v}|\mathcal{D})} \right) + \frac{N}{B} \sum_{i=1}^B \log p(y^{(i)}|\mathbf{x}^{(i)}, \mathbf{v}) \right]$$

where N is the size of the dataset \mathcal{D} and $\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^B \subset \mathcal{D}$ is a minibatch of size $B < N$. We use the standard train and test splits of both datasets. See Table 2 for classification accuracies and log-likelihoods reported on the train and test sets of each dataset.

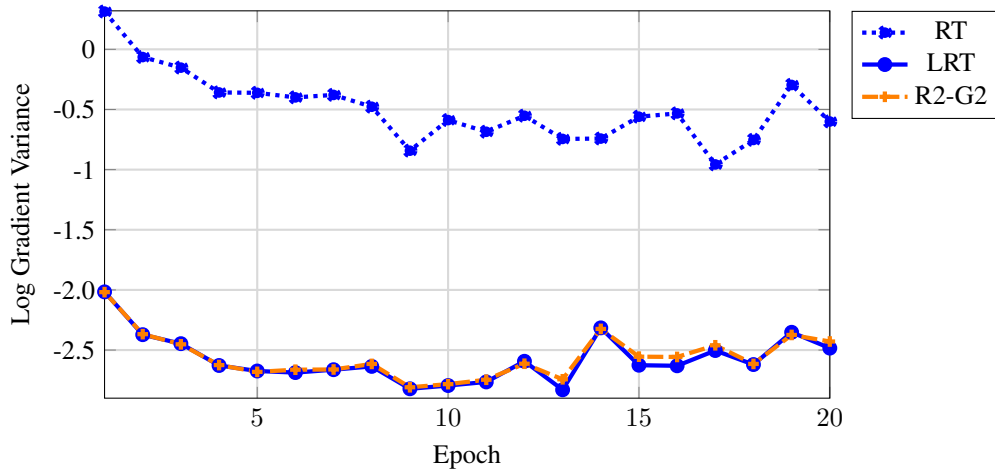


Figure 1: Log average gradient variance v.s. epoch for the top layer of a Bayesian MLP trained on *MNIST* over 5 runs. We compare the variance of gradients when training using the reparameterisation (RT), local reparameterisation (LRT) and R2-G2 estimators.

For *MNIST*, we used the two-layer multi-layer perceptron (MLP) architecture from Srivastava et al. (2014), which consists of two hidden layers of 1024 ReLU units. Each estimator is applied to all linear layers. For experiments with the R2-G2 estimator, we compute pre-activations in the same way as the local reparameterisation trick (i.e. sampling pre-activations directly). Our results illustrate that accuracies and log-likelihoods for the R2-G2 and LRT estimators match exactly. In Figures 1 and 3, the variance of gradients for the R2-G2 and LRT estimators also coincide and are much lower than those of the RT estimator. These observations empirically supports Theorem 4.3: the LRT estimator is an instance of the R2-G2 estimator and Rao-Blackwellisation. In other words, the local reparameterisation trick is equivalent to directly sampling pre-activations in forward computations and using a Rao-Blackwellised RT estimator in backpropagation.

For *CIFAR-10*, we used the VGG-11 architecture described in Simonyan and Zisserman (2015), where each estimator is applied to the last four convolutional layers. For the last four convolutional layers, we do not include the local reparameterisation trick as a benchmark since sampling *scalar* pre-activations directly would ignore dependencies induced by sharing weights in a convolutional layer (i.e. *additionally assumes* $\mathbf{A}\mathbf{A}^\top$ is diagonal). The RT estimator is applied to the first four convolutional layers. To limit the effect of gradient variances from linear layers on gradient variances of convolutional layers, we applied the LRT estimator to all linear layers.

Across both datasets, we found that initial training with the R2-G2 estimator yields higher log-likelihoods than the RT estimator while enjoying similar levels of accuracy. These results extend the benefits of training with Rao-Blackwellised gradients from Bayesian MLPs to Bayesian CNNs.

5.3 Generative modelling with Hierarchical Variational Autoencoders

We consider VAEs for generative modelling tasks on three standard benchmark datasets: *MNIST* (LeCun et al., 2010), *Omniglot* (Lake et al., 2015) and *Fashion-MNIST* (Xiao et al., 2017). We do not focus on comparisons of gradient variances due to architecture constraints of VAEs. Validating Proposition 4.2 requires independent and identically distributed (i.i.d.) samples of $\mathbf{z} \sim \tilde{q}_{\mathbf{z}}$ which requires computing a Cholesky factor in each iteration. This is computationally expensive and not representative of computations in VAEs. In practice, we compute $\mathbf{z} = \mathbf{W} \cdot g(\epsilon, \theta)$ where \mathbf{W} is the linear layer following reparameterisation, which does not enable sampling the full support of $\tilde{q}_{\mathbf{z}}$.

We found that training one-layer VAEs with the R2-G2 estimator did not guarantee performance gains (see Appendix H), and surmise that reducing gradient variance hinders the learning of stable

Table 3: Test variational lower bounds for hierarchical VAEs using the R2-G2 and Reparameterisation (RT) estimators. Higher is better. Error bars denote ± 1.96 standard errors ($\sigma/\sqrt{5}$) over 5 runs. See text for details.

# VAE Layers	Estimator	MNIST	Omniglot	Fashion-MNIST
2	R2-G2	-106.85 ± 5.00	-129.80 ± 0.74	-240.23 ± 0.64
	RT	-107.64 ± 8.46	-131.48 ± 1.66	-240.59 ± 0.93
3	R2-G2	-102.45 ± 3.73	-134.95 ± 2.26	-240.15 ± 0.86
	RT	-111.50 ± 5.40	-136.12 ± 2.70	-240.89 ± 0.97

representations in this setting as only training of the encoder is affected by the R2-G2 estimator. An investigation of this phenomena is beyond the scope of this paper and we leave this for future work.

We present the results of our experiments for two-layer and three-layer VAEs (i.e. hierarchical VAEs). In this setting, the input data \mathcal{D} is an observation and the latent variables \mathbf{v} is its latent representation (i.e. \mathbf{v} are local variables). The objective is to maximise the variational lower bound on the log-likelihood

$$\log p(\mathbf{x}) > \mathbb{E}_{q_{\theta}(\mathbf{v}^{(1)}|\mathcal{D}), \dots, q_{\theta}(\mathbf{v}^{(K)}|\mathcal{D})} \left[\log \left(\frac{1}{K} \sum_{i=1}^K \frac{p(\mathcal{D}, \mathbf{v}^{(i)})}{q_{\theta}(\mathbf{v}^{(i)}|\mathcal{D})} \right) \right]$$

where \mathcal{D} denotes input data, and $\{\mathbf{v}^{(i)}\}_{i=1}^K$ are vectors of latent Gaussian random variables. For training, we use a single sample ($K = 1$), which is equivalent to variational inference. For testing, we use 5000 samples ($K = 5000$), which is equivalent to importance weighted variational inference, providing a tighter bound on the log-likelihood (Burda et al., 2016). For each L -layer VAE, we used a bottom-up variational posterior and top-down generative process with latent variables $\{\mathbf{v}_i\}_{i=1}^L$

$$q(\{\mathbf{v}_i\}_{i=1}^L|\mathbf{x}) = q(\mathbf{v}_1|\mathbf{x}) \prod_{l=2}^L q(\mathbf{v}_l|\mathbf{v}_{l-1}, \mathbf{x}), \quad p(\mathbf{x}, \{\mathbf{v}_i\}_{i=1}^L) = p(\mathbf{x}|\{\mathbf{v}_i\}_{i=1}^L) \prod_{l=1}^{L-1} p(\mathbf{v}_l|\mathbf{v}_{l+1})$$

with latent spaces of 50 units and each conditional distribution is parameterised by a MLP with two hidden layers of 200 \tanh units (see Burda et al. (2016); Bauer and Mnih (2021)). We used a factorised Bernoulli likelihood, and factorised Gaussian variational posterior and prior.

We applied the R2-G2 estimator as a single layer within the *decoder* that applies reparameterisation and the linear transformation that follows it (i.e. \mathbf{W} and \mathbf{V} are decoder parameters). We used dynamic binarisation of all three datasets (Salakhutdinov and Murray, 2008). We used the standard train and test splits for MNIST and Fashion-MNIST, and the train and test split from (Burda et al., 2016) for Omniglot. See Table 3 for test lower bounds on the log-likelihood of all datasets after 100,000 steps.

Across all datasets and hierarchical VAEs, we found that the R2-G2 estimator consistently yielded higher test ELBOs than the RT estimator. For three-layer VAEs, we saw substantial gains of 9.05 and 1.17 nats on MNIST and Omniglot respectively (see Figures 2 and 9). These results extend benefits of initial training with Rao-Blackwellised gradients, from Bayesian NNs to Hierarchical VAEs.

6 Conclusion

We have presented the R2-G2 estimator as a novel, general-purpose and single-sample gradient estimator for latent Gaussian random variables as the Rao-Blackwellisation of the reparameterisation gradient estimator. Our method is motivated by the widespread usage of Gaussian distributions in probabilistic ML and applications of Rao-Blackwellisation in gradient estimators for latent discrete variables (Liu et al., 2019; Dong et al., 2020; Kool et al., 2020; Paulus et al., 2021).

We theoretically and empirically show that the local reparameterisation trick is an instance of Rao-Blackwellisation and the R2-G2 estimator for linear layers of BNNs with independent weights. It is equivalent to sampling pre-activations in forward computations and updating parameters with a Rao-Blackwellised reparameterisation gradient estimator in backpropagation. This explicates the empirical evidence that the local reparameterisation trick reduces variance of gradients obtained

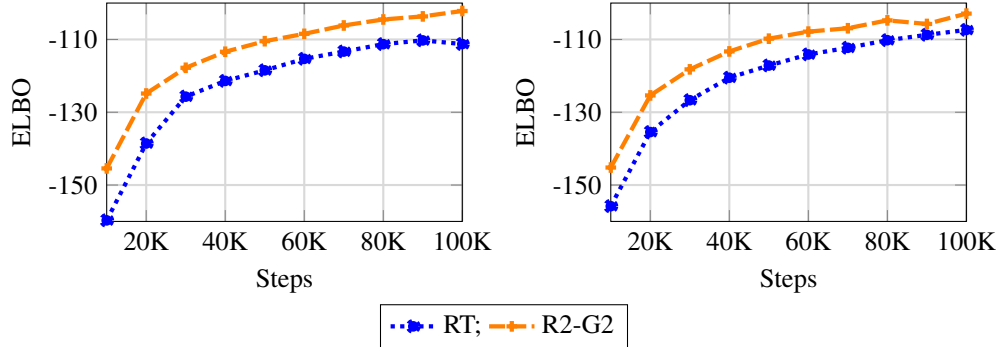


Figure 2: Bounds on log-likelihood v.s. optimisation steps for a three-layer VAE trained on *MNIST* over 5 runs. We compare the bounds on log-likelihoods when training using the reparameterisation (RT) and R2-G2 estimators. Training with the R2-G2 estimator improves bounds on log-likelihood on both the training set (left) and test set (right).

by the global reparameterisation trick as the benefit of Rao-Blackwellisation stated by Theorem 4.3. By casting the local reparameterisation gradient estimator as Rao-Blackwellised gradients for Bayesian MLPs, we showed that initial training with Rao-Blackwellised gradients also yield gains in performance for other models such as Bayesian CNNs and hierarchical VAEs. While performance gains were sometimes modest, they were consistent across models with multiple applications of the reparameterisation trick and particularly prominent in hierarchical VAEs. The main limitation of the R2-G2 estimator is the computational cost from solving Equation 9 and its implicit dependence on how pre-activations are computed. The latter is inherited from model parameterisation. To mitigate the former, we exploited low dimensionality in upper convolutional layers of Bayesian CNNs and latent spaces of VAEs, thereby requiring less iterations of the conjugate gradient algorithm.

While our work has focused on the case where the matrix \mathbf{V} is diagonal, we note that R2-G2 can be extended to the non-diagonal case, such as low-rank covariance matrices (Tomczak et al., 2020), by defining appropriate matrix-vector product functions called within the conjugate gradient algorithm. Aside from likelihoods and ELBOs, we note that the R2-G2 estimator can be applied to other objective functions. A potential extension of our work can be to augment existing gradient estimators for variational inference in the *multi-sample* setting (Roeder et al., 2017; Rainforth et al., 2018; Tucker et al., 2019; Bauer and Mnih, 2021) with the R2-G2 estimator and evaluate its effectiveness. We leave these directions for future work.

7 Acknowledgements

We would like to thank Andriy Mnih for helpful discussions and comments on drafts of this paper. Kevin H. Lam gratefully acknowledges his PhD funding from Google DeepMind. Thang D. Bui acknowledges support from the National Computing Infrastructure (NCI) Australia. George Deligianidis acknowledges support from the Engineering and Physical Sciences Research Council [grant number EP/Y018273/1]. Yee Whye Teh acknowledges support from the Ministry of Digital Development and Information (MDDI) under the Singapore Global AI Visiting Professorship Program (Award No. AIVP-2024-002).

References

- Matthias Bauer and Andriy Mnih. Generalized doubly reparameterized gradient estimators. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 738–747. PMLR, 18–24 Jul 2021.
- David Blackwell. Conditional expectation and unbiased sequential estimation. *The Annals of Mathematical Statistics*, 18(1):105–110, 1947.
- Yuri Burda, Roger B. Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. In *International Conference on Learning Representations*, 2016.

- Zhe Dong, Andriy Mnih, and George Tucker. Disarm: An antithetic gradient estimator for binary latent variables. In *Advances in Neural Information Processing Systems*, volume 33, pages 18637–18647. Curran Associates, Inc., 2020.
- Morris L. Eaton. *Multivariate statistics : a vector space approach*. Wiley series in probability and mathematical statistics. Probability and mathematical statistics. Wiley, New York, 1983. ISBN 0471027766.
- Mikhail Figurnov, Shakir Mohamed, and Andriy Mnih. Implicit reparameterization gradients. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.
- Paul Glasserman. *Monte Carlo methods in financial engineering*, volume 53. Springer, 2003.
- Peter W. Glynn. Likelihood ratio gradient estimation for stochastic systems. *Communications of the ACM*, 33(10):75–84, October 1990. ISSN 0001-0782.
- Evan Greensmith, Peter Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001.
- Ken Hayami. Convergence of the conjugate gradient method on singular systems. *arXiv preprint arXiv:1809.00793*, 2018.
- Chin-Wei Huang, Ricky T. Q. Chen, Christos Tsirigotis, and Aaron Courville. Convex potential flows: Universal probability distributions with optimal transport and convex optimization. In *International Conference on Learning Representations*, 2021.
- Iris A. M. Huijben, Wouter Kool, Max B. Paulus, and Ruud J. G. van Sloun. A review of the Gumbel-max trick and its extensions for discrete stochasticity in machine learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):1353–1371, 2023.
- M. James. The generalised inverse. *The Mathematical Gazette*, 62(420):109–114, 1978. ISSN 00255572, 20566328.
- Martin Jankowiak and Fritz Obermeyer. Pathwise derivatives beyond the reparameterization trick. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2235–2244. PMLR, 10–15 Jul 2018.
- Erik F Kaasschieter. Preconditioned conjugate gradients for solving singular systems. *Journal of Computational and Applied mathematics*, 24(1-2):265–275, 1988.
- Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.
- Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.
- Durk P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.
- Wouter Kool, Herke van Hoof, and Max Welling. Estimating gradients for discrete random variables by sampling without replacement. In *International Conference on Learning Representations*, 2020.
- Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, Toronto, Ontario, 2009.
- Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.
- Yann LeCun, Corinna Cortes, and CJ Burges. MNIST handwritten digit database. *ATT Labs*, 2, 2010.
- Runjing Liu, Jeffrey Regier, Nilesch Tripurani, Michael Jordan, and Jon McAuliffe. Rao-Blackwellized stochastic gradients for discrete distributions. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4023–4031. PMLR, 09–15 Jun 2019.

- Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte Carlo gradient estimation in machine learning. *Journal of Machine Learning Research*, 21(132):1–62, 2020.
- Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, 1999. ISBN 978-0-387-98793-4.
- Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.
- Max B. Paulus, Chris J. Maddison, and Andreas Krause. Rao-Blackwellizing the straight-through gumbel-softmax gradient estimator. In *International Conference on Learning Representations*, 2021.
- M. Planitz. Inconsistent systems of linear equations. *The Mathematical Gazette*, 63(425):181–185, 1979. ISSN 00255572, 20566328.
- Robert Price. A useful theorem for nonlinear devices having Gaussian inputs. *IRE Transactions on Information Theory*, 4(2):69–72, 1958.
- Tom Rainforth, Adam Kosiorek, Tuan Anh Le, Chris Maddison, Maximilian Igl, Frank Wood, and Yee Whye Teh. Tighter variational bounds are not necessarily better. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4277–4285. PMLR, 10–15 Jul 2018.
- C Radhakrishna Rao et al. Information and the accuracy attainable in the estimation of statistical parameters. *Breakthroughs in statistics*, pages 235–247, 1992.
- Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1278–1286, Beijing, China, 22–24 Jun 2014. PMLR.
- Geoffrey Roeder, Yuhuai Wu, and David K Duvenaud. Sticking the landing: Simple, lower-variance gradient estimators for variational inference. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
- Ruslan Salakhutdinov and Iain Murray. On the quantitative analysis of deep belief networks. In *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, volume 307 of *ACM International Conference Proceeding Series*, pages 872–879. ACM, 2008.
- Tim Salimans. Implementing and automating fixed-form variational posterior approximation through stochastic linear regression. *arXiv preprint arXiv:1401.2135*, 2014.
- Tim Salimans and David A Knowles. Fixed-form variational posterior approximation through stochastic linear regression. *Bayesian Analysis*, 8(4):837–882, 2013.
- Tim Salimans and David A Knowles. On using control variates with stochastic approximation for variational Bayes and its connection to stochastic linear regression. *arXiv preprint arXiv:1401.1022*, 2014.
- Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.
- Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

- Michalis Titsias and Miguel Lázaro-Gredilla. Doubly stochastic variational Bayes for non-conjugate inference. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1971–1979, Beijing, China, 22–24 Jun 2014. PMLR.
- Marcin Tomczak, Siddharth Swaroop, and Richard Turner. Efficient low rank gaussian variational inference for neural networks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 4610–4622. Curran Associates, Inc., 2020.
- George Tucker, Dieterich Lawson, Shixiang Gu, and Chris J. Maddison. Doubly reparameterized gradient estimators for Monte Carlo objectives. In *International Conference on Learning Representations*, 2019.
- Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.
- Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747, 2017.
- Ming Xu, Matias Quiroz, Robert Kohn, and Scott A. Sisson. Variance reduction properties of the reparameterization trick. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 2711–2720. PMLR, 16–18 Apr 2019.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: Our main theoretical and experimental contributions are clearly stated in the abstract and demonstrated in the paper. They reflect the paper's contributions and scope.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: We discuss the limitations of our work in Section 6.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[Yes\]](#)

Justification: All proofs are provided in the technical appendices and appropriately referenced.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [\[Yes\]](#)

Justification: We provide full experiment details in Appendix E and the code to run our experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide the code needed to run our experiments.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We provide full experiment details in Appendix E.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We report error bars in all tabulated results from our experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We provide full details of compute resources in Appendix E.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: After careful review of the NeurIPS Code of Ethics, it is clear that the research conducted in this paper conforms with the Code of Ethics in every respect.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: This paper is mostly methodological. While it is possible for advances in machine learning to bring societal impacts, our paper remains general and not specific to any application so it is unlikely to bring about any immediate societal impact.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This paper poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All datasets used in our experiments are open-source and have been properly referenced in the paper and Appendix J.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: This paper does not introduce new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: This paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: This paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: This paper does not involve usage of LLMs.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.

A Supporting results

This section lists the results supporting the design of the R2-G2 estimator, namely the analytical form of conditional Gaussian distributions and the conjugate gradient algorithm.

Lemma A.1 ((Eaton, 1983, Proposition 3.13)) Suppose we have a random vector $\epsilon \sim \mathcal{N}(\mathbf{0}_n, \mathbf{I}_n)$ where $\epsilon \in \mathbb{R}^n$ and a linear map defined as

$$\mathbf{A} : \mathbb{R}^n \rightarrow \mathbb{R}^m, \quad \mathbf{A}(\mathbf{x}) = \mathbf{A}\mathbf{x}$$

where $\mathbf{A} \in \mathbb{R}^{m \times n}$. Then we have the joint distribution

$$\begin{bmatrix} \epsilon \\ \mathbf{A}(\epsilon) \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} \mathbf{0}_n \\ \mathbf{0}_m \end{bmatrix}, \begin{bmatrix} \mathbf{I}_n & \mathbf{A}^\top \\ \mathbf{A} & \mathbf{A}\mathbf{A}^\top \end{bmatrix} \right),$$

and the conditional distribution

$$\epsilon | \mathbf{A}(\epsilon) = \mathbf{z} \sim \mathcal{N} \left(\mathbf{A}^\top (\mathbf{A}\mathbf{A}^\top)^\dagger \mathbf{z}, \mathbf{I}_n - \mathbf{A}^\top (\mathbf{A}\mathbf{A}^\top)^\dagger \mathbf{A} \right)$$

where $(\mathbf{A}\mathbf{A}^\top)^\dagger$ is the Moore-Penrose pseudo-inverse of $\mathbf{A}\mathbf{A}^\top$.

Algorithm 2 Conjugate Gradient Algorithm

Input: number of iterations T , matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, column vector $\mathbf{z} \in \mathbb{R}^m$.

Initialise $\mathbf{x}_0 \in \mathbb{R}^n$.

Set $\mathbf{r}_0 = \mathbf{A}\mathbf{A}^\top \mathbf{x}_0 - \mathbf{z}$.

Set $\mathbf{p}_0 = -\mathbf{r}_0$.

Set $k = 0$.

while $k < T$ and $\mathbf{r}_k \neq \mathbf{0}$ **do**

 Compute $\alpha_k = \frac{\mathbf{r}_k^\top \mathbf{r}_k}{\mathbf{p}_k^\top \mathbf{A}\mathbf{A}^\top \mathbf{p}_k}$.

 Compute $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$.

 Compute $\mathbf{r}_{k+1} = \mathbf{r}_k + \alpha_k \mathbf{A}\mathbf{A}^\top \mathbf{p}_k$.

 Compute $\beta_{k+1} = \frac{\mathbf{r}_{k+1}^\top \mathbf{r}_{k+1}}{\mathbf{r}_k^\top \mathbf{r}_k}$.

 Compute $\mathbf{p}_{k+1} = -\mathbf{r}_{k+1} + \beta_{k+1} \mathbf{p}_k$.

 Increment k by 1.

end while

Output: \mathbf{x}_T .

B Proof for Proposition 4.2

We first prove that the R2-G2 estimator is an unbiased estimator of $\nabla_{\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\theta}}}[\ell]$. Recall that

$$\widehat{\nabla_{\boldsymbol{\theta}} \ell}^{R2-G2} = \mathbb{E}_{\tilde{q}_0} \left[\widehat{\nabla_{\boldsymbol{\theta}} \ell}^{RT} \right].$$

We note that $\widehat{\nabla_{\boldsymbol{\theta}} \ell}^{R2-G2}$ is a conditional expectation with $\mathbf{W} \cdot g = \mathbf{z}$ as the conditioning variable. Then we have

$$\begin{aligned} \mathbb{E}_{q_{\mathbf{z}}} \left[\widehat{\nabla_{\boldsymbol{\theta}} \ell}^{R2-G2} \right] &= \mathbb{E}_{q_{\mathbf{z}}} \left[\mathbb{E}_{\tilde{q}_0} \left[\widehat{\nabla_{\boldsymbol{\theta}} \ell}^{RT} \right] \right] \\ &= \mathbb{E}_{q_0} \left[\widehat{\nabla_{\boldsymbol{\theta}} \ell}^{RT} \right] \\ &= \nabla_{\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\theta}}}[\ell] \end{aligned}$$

where the second equality follows from the law of iterated expectations. Since the reparameterisation gradient estimator $\widehat{\nabla_{\boldsymbol{\theta}} \ell}^{RT}$ is an unbiased estimator of $\nabla_{\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\theta}}}[\ell]$, it follows that the R2-G2 estimator $\widehat{\nabla_{\boldsymbol{\theta}} \ell}^{R2-G2}$ is also an unbiased estimator of $\nabla_{\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\theta}}}[\ell]$.

We are left to show that the R2-G2 estimator has less variance than the reparameterisation gradient estimator. We can write

$$\begin{aligned} \mathbb{E}_{q_{\mathbf{z}}} \left[\left\| \widehat{\nabla_{\boldsymbol{\theta}} \ell}^{R2-G2} - \nabla_{\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\theta}}}[\ell] \right\|^2 \right] &= \mathbb{E}_{q_{\mathbf{z}}} \left[\left\| \mathbb{E}_{\tilde{q}_0} \left[\widehat{\nabla_{\boldsymbol{\theta}} \ell}^{RT} \right] - \nabla_{\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\theta}}}[\ell] \right\|^2 \right] \\ &= \mathbb{E}_{q_{\mathbf{z}}} \left[\left\| \mathbb{E}_{\tilde{q}_0} \left[\widehat{\nabla_{\boldsymbol{\theta}} \ell}^{RT} - \nabla_{\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\theta}}}[\ell] \right] \right\|^2 \right] \\ &\leq \mathbb{E}_{q_{\mathbf{z}}} \left[\mathbb{E}_{\tilde{q}_0} \left[\left\| \widehat{\nabla_{\boldsymbol{\theta}} \ell}^{RT} - \nabla_{\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\theta}}}[\ell] \right\|^2 \right] \right] \\ &= \mathbb{E}_{q_0} \left[\left\| \widehat{\nabla_{\boldsymbol{\theta}} \ell}^{RT} - \nabla_{\boldsymbol{\theta}} \mathbb{E}_{q_{\boldsymbol{\theta}}}[\ell] \right\|^2 \right] \end{aligned}$$

where the inequality results from using Jensen's inequality, and the last equality comes from the law of iterated expectations.

C Proof of Theorem 4.3

Suppose we have a BNN linear layer where weights are independent Gaussian random variables. Given an input $\mathbf{x} \in \mathbb{R}^n$, the pre-activations and parameters of these linear layers are respectively given by

$$\mathbf{z} = \begin{bmatrix} \mathbf{x}^\top g(\boldsymbol{\epsilon}^{(1)}, \boldsymbol{\theta}^{(1)}) \\ \vdots \\ \mathbf{x}^\top g(\boldsymbol{\epsilon}^{(m)}, \boldsymbol{\theta}^{(m)}) \end{bmatrix} \in \mathbb{R}^m$$

where $\boldsymbol{\theta}^{(i)} = \{\boldsymbol{\mu}^{(i)}, \boldsymbol{\tau}^{(i)}\}$ for $i = 1, \dots, m$. We proceed by considering a fixed i .

By setting $\mathbf{V}^{(i)} = (\boldsymbol{\Sigma}^{(i)})^{\frac{1}{2}} = (\text{diag}(\boldsymbol{\tau}^{(i)}))^{\frac{1}{2}}$ and $\mathbf{W} = \mathbf{x}^\top$ within Equation 6, the reparameterisation gradient estimator is given by

$$\widehat{\nabla_{\boldsymbol{\theta}^{(i)}} \ell}^{RT} = \left(\frac{\partial \tilde{\ell}}{\partial z_i} \cdot \mathbf{x}^\top \cdot \left[\mathbf{I}_n \mid \frac{1}{2} (\boldsymbol{\Sigma}^{(i)})^{-\frac{1}{2}} \odot (\mathbf{1}_n (\boldsymbol{\epsilon}^{(i)})^\top) \right] \right)^\top.$$

We proceed with the idea of Rao-Blackwellisation from the R2-G2 estimator by conditioning on the pre-activation $z_i = \mathbf{x}^\top \cdot g(\boldsymbol{\epsilon}^{(i)}, \boldsymbol{\theta}^{(i)}) \in \mathbf{z}$. This amounts to setting $\mathbf{A} = \mathbf{x}^\top \mathbf{V}^{(i)}$ in Definition 4.1.

We note that $\mathbf{x}^\top \cdot g(\boldsymbol{\epsilon}^{(i)}, \boldsymbol{\theta}^{(i)}) = z_i$ is equivalent to $\mathbf{A} \boldsymbol{\epsilon}^{(i)} = z_i - \mathbf{x}^\top \boldsymbol{\mu}^{(i)} = \tilde{z}_i$. Denote $\tilde{\boldsymbol{\epsilon}}^{(i)}$ as $\boldsymbol{\epsilon}^{(i)} | \mathbf{A} \boldsymbol{\epsilon}^{(i)} = \tilde{z}_i$ with distribution denoted $\tilde{q}_0^{(i)}$. Using Lemma A.1, we can write the distribution

$\tilde{\boldsymbol{\epsilon}}^{(i)} \sim \tilde{q}_0^{(i)}$ as

$$\tilde{q}_0^{(i)} = \mathcal{N} \left((\boldsymbol{\Sigma}^{(i)})^{\frac{1}{2}} \mathbf{x} \left(\frac{\tilde{z}_i}{\sum_{j=1}^n x_j^2 (\sigma_j^{(i)})^2} \right), \mathbf{I}_n - (\boldsymbol{\Sigma}^{(i)})^{\frac{1}{2}} \mathbf{x} \left(\sum_{j=1}^n x_j^2 (\sigma_j^{(i)})^2 \right)^{-1} \mathbf{x}^\top (\boldsymbol{\Sigma}^{(i)})^{\frac{1}{2}} \right).$$

Define the transformation of a vector to a diagonal matrix as

$$D : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times n}, \quad D(\mathbf{x}) = \sum_{k=1}^n e_k^\top \mathbf{x} e_k e_k^\top$$

where $e_k \in \mathbb{R}^n$ is the k -th standard basis vector. Using the linearity of expectations, we have

$$\begin{aligned} \frac{1}{2} \mathbf{x}^\top \cdot \mathbb{E}_{\tilde{q}_0^{(i)}} \left[(\boldsymbol{\Sigma}^{(i)})^{-\frac{1}{2}} \odot \mathbf{1}_n (\boldsymbol{\epsilon}^{(i)})^\top \right] &= \frac{1}{2} \mathbf{x}^\top \cdot \left[(\boldsymbol{\Sigma}^{(i)})^{-\frac{1}{2}} \odot \mathbf{1}_n \left(\mathbb{E}_{\tilde{q}_0^{(i)}} [\boldsymbol{\epsilon}^{(i)}] \right)^\top \right] \\ &= \frac{1}{2} \left(\frac{\tilde{z}_i}{\sum_{j=1}^n x_j^2 (\sigma_j^{(i)})^2} \right) \mathbf{x}^\top \cdot \left[(\boldsymbol{\Sigma}^{(i)})^{-\frac{1}{2}} \odot (\mathbf{1}_n \mathbf{x}^\top (\boldsymbol{\Sigma}^{(i)})^{\frac{1}{2}}) \right] \\ &= \frac{1}{2} \left(\frac{\tilde{z}_i}{\sum_{j=1}^n x_j^2 (\sigma_j^{(i)})^2} \right) \mathbf{x}^\top D(\mathbf{x}) \\ &= \frac{1}{2} \left(\frac{z_i - \sum_{j=1}^n x_j \mu_j^{(i)}}{\sum_{j=1}^n x_j^2 (\sigma_j^{(i)})^2} \right) (\mathbf{x} \odot \mathbf{x})^\top \\ &\stackrel{d}{=} \frac{1}{2} \left(\sum_{j=1}^n x_j^2 (\sigma_j^{(i)})^2 \right)^{-\frac{1}{2}} \xi_i (\mathbf{x} \odot \mathbf{x})^\top \end{aligned}$$

where $\mathbf{x}^\top D(\mathbf{x}) = (\mathbf{x} \odot \mathbf{x})^\top$ is a row vector with entries $\{x_j^2\}_{j=1}^n$ and applying the reparameterisation trick gives us

$$z_i \stackrel{d}{=} \sum_{j=1}^n x_j \mu_j^{(i)} + \left(\sum_{j=1}^n x_j^2 (\sigma_j^{(i)})^2 \right)^{\frac{1}{2}} \xi_i \sim \mathcal{N} \left(\sum_{j=1}^n x_j \mu_j^{(i)}, \sum_{j=1}^n x_j^2 (\sigma_j^{(i)})^2 \right)$$

for $\xi_i \sim \mathcal{N}(0, 1)$. Since we also have $\mathbf{x}^\top \cdot \mathbb{E}_{\tilde{q}_0^{(i)}} [\mathbf{I}_n] = \mathbf{x}^\top$, the Jacobian of $z_i = \mathbf{x}^\top \cdot g(\boldsymbol{\epsilon}^{(i)}, \boldsymbol{\theta}^{(i)})$ is

$$\begin{aligned} J_{z_i}(\boldsymbol{\theta}^{(i)}) &= \mathbf{x}^\top \cdot \mathbb{E}_{\tilde{q}_0^{(i)}} [J_{g^{(i)}}] \\ &\stackrel{d}{=} \left[\mathbf{x}^\top \mid \frac{1}{2} \left(\sum_{j=1}^n x_j^2 (\sigma_j^{(i)})^2 \right)^{-\frac{1}{2}} \xi_i (\mathbf{x} \odot \mathbf{x})^\top \right]. \end{aligned}$$

Hence, we have

$$\widehat{\nabla_{\boldsymbol{\theta}^{(i)}} \ell}^{R2-G2} \stackrel{d}{=} \widehat{\nabla_{\boldsymbol{\theta}^{(i)}} \ell}^{LRT}.$$

This shows that the local reparameterisation gradient estimator is equivalent in distribution to a Rao-Blackwellised reparameterisation gradient estimator.

We are left to show that the local reparameterisation estimator has lower variance than the global reparameterisation estimator. We have

$$\begin{aligned}
\mathbb{E}_{\tilde{q}_{z_i}} \left[\left\| \widehat{\nabla_{\boldsymbol{\theta}^{(i)}} \ell}^{LRT} - \nabla_{\boldsymbol{\theta}^{(i)}} \mathbb{E}_{q_{\boldsymbol{\theta}^{(i)}}} [\ell] \right\|^2 \right] &= \mathbb{E}_{\tilde{q}_{z_i}} \left[\left\| \widehat{\nabla_{\boldsymbol{\theta}^{(i)}} \ell}^{R2-G2} - \nabla_{\boldsymbol{\theta}^{(i)}} \mathbb{E}_{q_{\boldsymbol{\theta}^{(i)}}} [\ell] \right\|^2 \right] \\
&= \mathbb{E}_{\tilde{q}_{z_i}} \left[\left\| \mathbb{E}_{\tilde{q}_0} \left[\widehat{\nabla_{\boldsymbol{\theta}^{(i)}} \ell}^{RT} \right] - \nabla_{\boldsymbol{\theta}^{(i)}} \mathbb{E}_{q_{\boldsymbol{\theta}^{(i)}}} [\ell] \right\|^2 \right] \\
&= \mathbb{E}_{\tilde{q}_{z_i}} \left[\left\| \mathbb{E}_{\tilde{q}_0} \left[\widehat{\nabla_{\boldsymbol{\theta}^{(i)}} \ell}^{RT} - \nabla_{\boldsymbol{\theta}^{(i)}} \mathbb{E}_{q_{\boldsymbol{\theta}^{(i)}}} [\ell] \right] \right\|^2 \right] \\
&\leq \mathbb{E}_{\tilde{q}_{z_i}} \left[\mathbb{E}_{\tilde{q}_0} \left[\left\| \widehat{\nabla_{\boldsymbol{\theta}^{(i)}} \ell}^{RT} - \nabla_{\boldsymbol{\theta}^{(i)}} \mathbb{E}_{q_{\boldsymbol{\theta}^{(i)}}} [\ell] \right\|^2 \right] \right] \\
&= \mathbb{E}_{q_0^{(i)}} \left[\left\| \widehat{\nabla_{\boldsymbol{\theta}^{(i)}} \ell}^{RT} - \nabla_{\boldsymbol{\theta}^{(i)}} \mathbb{E}_{q_{\boldsymbol{\theta}^{(i)}}} [\ell] \right\|^2 \right]
\end{aligned}$$

where the inequality results from using Jensen's inequality and the last equality comes from the law of iterated expectations.

D Computation of Conditional Mean

Recall that the image and kernel of a linear map $\mathbf{T} : \mathbb{R}^n \rightarrow \mathbb{R}^m$ are defined respectively as $\text{im}(\mathbf{T}) = \{\mathbf{T}\mathbf{v} : \mathbf{v} \in \mathbb{R}^n\}$ and $\ker(\mathbf{T}) = \{\mathbf{v} : \mathbf{T}\mathbf{v} = \mathbf{0}\}$. Since $\mathbf{A}\boldsymbol{\epsilon} \in \text{im}(\mathbf{A}) = \text{im}(\mathbf{A}\mathbf{A}^\top)$, we note that Equation 9 has solutions in the form

$$(\mathbf{A}\mathbf{A}^\top)^\dagger \mathbf{A}\boldsymbol{\epsilon} + (\mathbf{I}_m - (\mathbf{A}\mathbf{A}^\top)^\dagger \mathbf{A}\mathbf{A}^\top) \mathbf{y}$$

for any $\mathbf{y} \in \mathbb{R}^m$ (see James (1978); Planitz (1979)). Here, $(\mathbf{A}\mathbf{A}^\top)^\dagger \mathbf{A}\boldsymbol{\epsilon} \in \text{im}(\mathbf{A}\mathbf{A}^\top) = \text{im}(\mathbf{A})$ and $(\mathbf{I}_m - (\mathbf{A}\mathbf{A}^\top)^\dagger \mathbf{A}\mathbf{A}^\top) \mathbf{y} \in \ker(\mathbf{A}\mathbf{A}^\top) = \ker(\mathbf{A}^\top)$. That is, we can recast the matrix-vector product $(\mathbf{A}\mathbf{A}^\top)^\dagger \mathbf{A}\boldsymbol{\epsilon}$ as a solution to Equation 9, up to an additive term from $\ker(\mathbf{A}^\top)$. For any solution $\boldsymbol{\beta}^*$ to Equation 9, it then follows that

$$\mathbf{A}^\top \boldsymbol{\beta}^* = \mathbf{A}^\top (\mathbf{A}\mathbf{A}^\top)^\dagger \mathbf{A}\boldsymbol{\epsilon} = \boldsymbol{\epsilon}^*.$$

Table 4: Optimisation hyperparameters and running times for experiments.

Model	Estimator	Steps	Learning Rate	Steps/s	Run Time Limit (hours)
Bayesian MLP	R2-G2	15,000	0.0001	13.57	0.5
Bayesian MLP	LRT	15,000	0.0001	48.25	0.5
Bayesian MLP	RT	15,000	0.0001	10.52	0.5
Bayesian CNN	R2-G2	12,500	0.0001	0.68	10
Bayesian CNN	RT	12,500	0.0001	2.31	5
One-layer VAE	R2-G2	100,000	0.0003	47.96	1
One-layer VAE	RT	100,000	0.0003	127.66	1
Two-layer VAE	R2-G2	100,000	0.0003	20.22	1.5
Two-layer VAE	RT	100,000	0.0003	78.72	1.5
Three-layer VAE	R2-G2	100,000	0.0003	17.11	2.5
Three-layer VAE	RT	100,000	0.0003	48.41	2.5

E Experiment details

Compute resources All experiments were run on a single NVIDIA V100 GPU.

Architectures Model architectures for BNN and VAE experiments can be found in Section 5. We excluded batch normalisation and dropout layers from model architectures.

Optimisation We used a batch size of 80 and the Adam optimiser for all experiments Kingma and Ba (2015). We do not add regularisation such as weight decay, dropout or batch normalisation layers. Other optimisation parameters are listed in Table 4.

For consistency with work on gradient estimators, we report the number of optimisation/SGD steps. For BNN experiments on the MNIST and CIFAR-10 datasets, this is equivalent to training with a batch size of 80 for 20 epochs.

F Comparison of Computational Cost of Gradient Estimators

In general, the worst-case complexity of computational costs from the conjugate gradient (CG) algorithm is $\mathcal{O}(m^3)$ when one is attempting to invert a $m \times m$ dense matrix \mathbf{A} without knowing its structure. Here the worst-case would be running m iterations of CG with each iteration requiring m^2 flops for a matrix-vector product.

In the mean-field setting, which is the main focus of our work, we know additional structure about the matrix \mathbf{A} . Specifically, it can factorise as $\mathbf{A} = \mathbf{W}\mathbf{V}\mathbf{V}^\top\mathbf{W}^\top$ where $\mathbf{V} \in \mathbb{R}^{n \times n}$ is diagonal and $\mathbf{W} \in \mathbb{R}^{m \times n}$. Moreover, we know the rank of \mathbf{V} and the maximum rank of \mathbf{W} , so CG only needs to be run for $k = \min(m, n)$ iterations at most, by using the property of ranks that $\text{rank}(AB) \leq \min(\text{rank}(A), \text{rank}(B))$ for matrices A, B .

A forward pass for a linear layer using the global reparameterisation trick would use a total of $(2m + 1)n$ flops for matrix-vector products. Using the R2-G2 estimator runs at most k iterations of the CG algorithm with each iteration requiring $(2m + 1)n$ flops for matrix-vector products, so an additional $(2m + 1)nk$ flops would be incurred at most.

For BNNs, we would have $n > m^2$ since there are more weights than pre-activations, so the cost complexity would be $\mathcal{O}(n)$ for a forward pass using the global reparameterisation trick and an additional $\mathcal{O}(m^2n)$ for R2-G2. For VAEs, we would have $m > n$ since we map low-dimensional latents to a higher-dimensional space, so the cost complexity would be $\mathcal{O}(m)$ for a forward pass using the global reparameterisation trick and an additional $\mathcal{O}(mn^2)$ for R2-G2.

G Additional plots of gradient variance

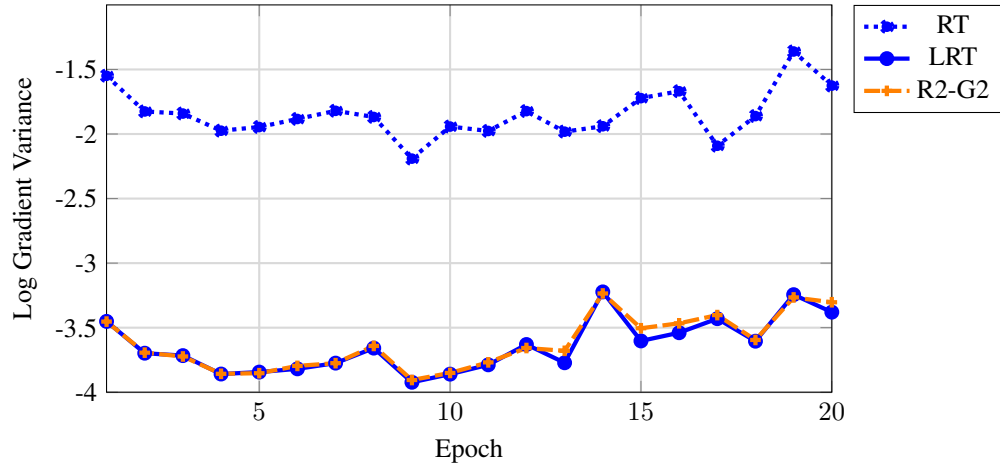


Figure 3: Log gradient variance v.s. epoch for the bottom layer of a Bayesian MLP trained on *MNIST* over 5 runs. We compare the variance of gradients when training using the reparameterisation (RT), local reparameterisation (LRT) and R2-G2 estimators.

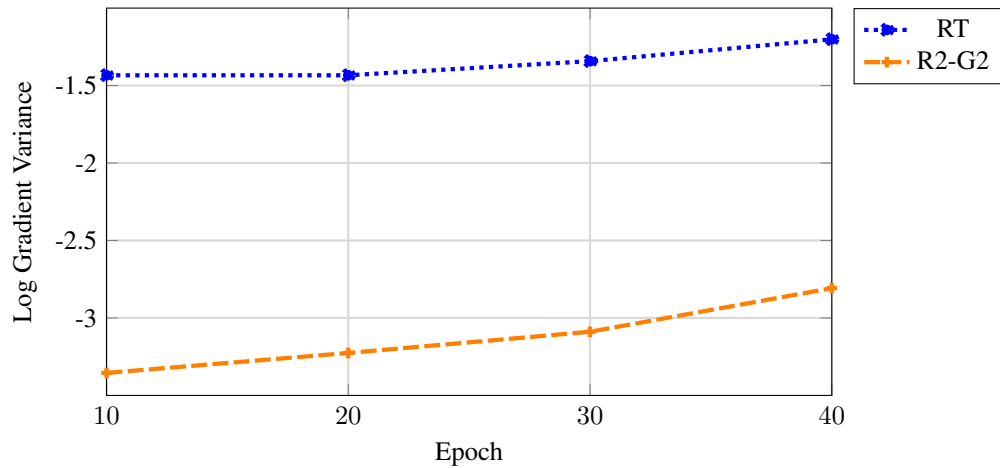


Figure 4: Log gradient variance v.s. epoch for the 8-th convolutional layer of a Bayesian CNN trained on *CIFAR-10* over 5 runs. We compare the variance of gradients when training using the reparameterisation (RT) and R2-G2 estimators.

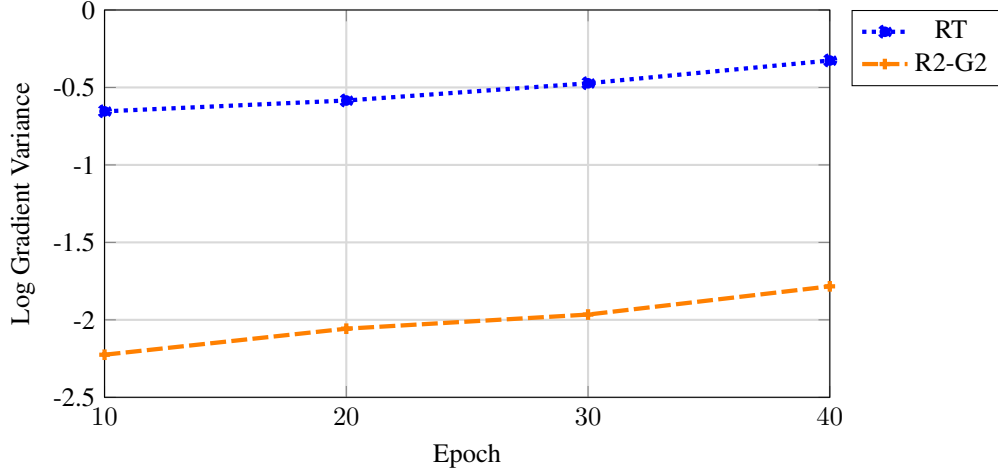


Figure 5: Log gradient variance v.s. epoch for the 5-th convolutional layer of a Bayesian CNN trained on *CIFAR-10* over 5 runs. We compare the variance of gradients when training using the reparameterisation (RT) and R2-G2 estimators.

H Experiments on single-layer Variational Autoencoders

Table 5: Test variational lower bounds for VAEs using the R2-G2 and Reparameterisation (RT) estimators. Higher is better. Error bars denote ± 1.96 standard errors ($\sigma/\sqrt{5}$) over 5 runs.

# VAE Layers	Estimator	MNIST	Omniglot	Fashion-MNIST
1	R2-G2	-94.39 ± 0.42	-117.61 ± 2.12	-238.65 ± 0.26
	RT	-94.22 ± 0.24	-117.64 ± 2.12	-238.64 ± 0.25

The R2-G2 estimator yielded limited gains on performance for single-layer VAEs. In this setting, the R2-G2 estimator only impacts the optimisation of the encoder. This motivates our experiments for hierarchical VAEs where we only apply the R2-G2 estimator within the decoder (i.e. \mathbf{W} and \mathbf{V} are both matrix parameters in the decoder).

I Additional plots of bounds on log-likelihood

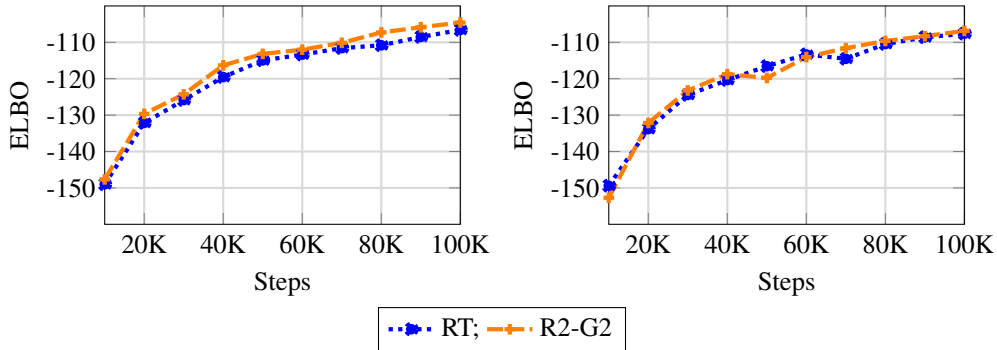


Figure 6: Bounds on log-likelihood v.s. optimisation steps for a two-layer VAE trained on *MNIST* over 5 runs. We compare the bounds on log-likelihoods when training using the reparameterisation (RT) and R2-G2 estimators on both the training set (left) and test set (right).

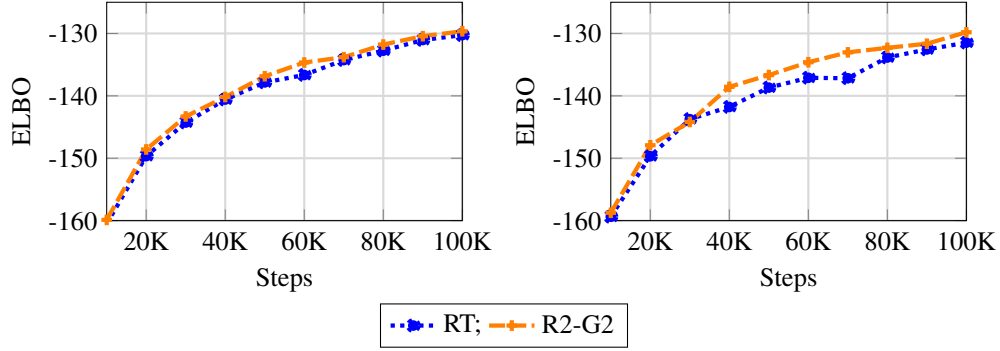


Figure 7: Bounds on log-likelihood v.s. optimisation steps for a two-layer VAE trained on *Omniglot* over 5 runs. We compare the bounds on log-likelihoods when training using the reparameterisation (RT) and R2-G2 estimators on both the training set (left) and test set (right).

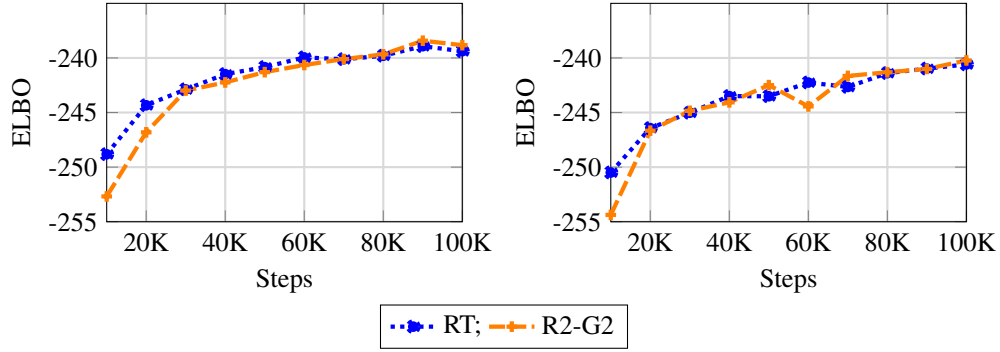


Figure 8: Bounds on log-likelihood v.s. optimisation steps for a two-layer VAE trained on *Fashion-MNIST* over 5 runs. We compare the bounds on log-likelihoods when training using the reparameterisation (RT) and R2-G2 estimators on both the training set (left) and test set (right).

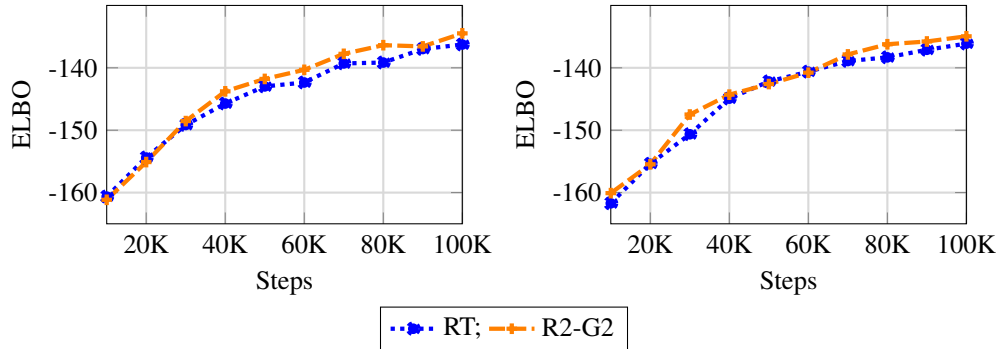


Figure 9: Bounds on log-likelihood v.s. optimisation steps for a three-layer VAE trained on *Omniglot* over 5 runs. We compare the bounds on log-likelihoods when training using the reparameterisation (RT) and R2-G2 estimators on both the training set (left) and test set (right).

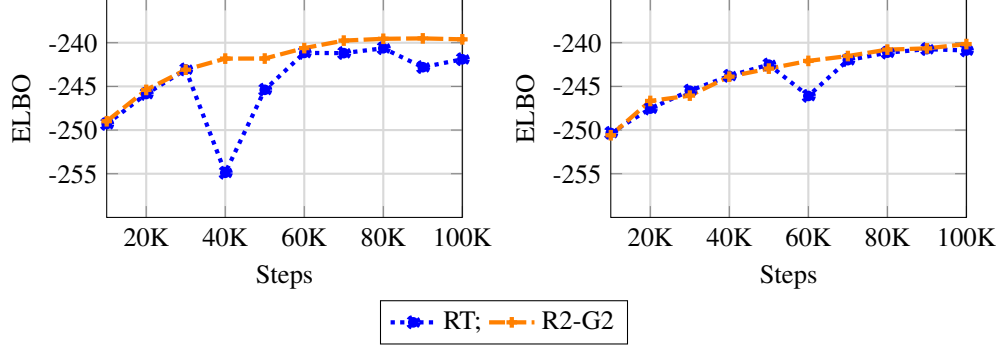


Figure 10: Bounds on log-likelihood v.s. optimisation steps for a three-layer VAE trained on *Fashion-MNIST* over 5 runs. We compare the bounds on log-likelihoods when training using the reparameterisation (RT) and R2-G2 estimators on both the training set (left) and test set (right).

J Licenses

Codebases:

- Convex Potential Flows: Universal Probability Distributions with Optimal Transport and Convex Optimization (Huang et al., 2021): MIT license.

Datasets:

- MNIST (LeCun et al., 2010): Creative Commons Attribution-Share Alike 3.0 license
- CIFAR-10 (Krizhevsky and Hinton, 2009): MIT license
- Fashion-MNIST (Xiao et al., 2017): MIT license
- Omniglot (Lake et al., 2015): MIT license

The conjugate gradient algorithm for each BNN and VAE architecture is modified from the above codebase. All experiments are performed on the above datasets.