Supplementary Material

A. Derivation of the Partially-Collapsed Variational Bound

In this section, we present the derivation of the partially-collapsed lower bound presented in Salimbeni et al. (2019). As in Section 2.1 in the main text, we consider the augmented joint distribution

$$p(y, f^{(1)}, f^{(2)}, z) = \prod_n^N p(y_n|f^{(1)}, f^{(2)}, z; x_n) \prod_{\ell=1}^2 p\left(f^{(\ell)}; f^{(\ell-1)}\right), \tag{A.1}$$

where $\{(x_n, y_n)\}_{n=1}^N$ are input output pairs. To perform variational inference in such a DGP model with latent variables, we can maximize an evidence lower bound (ELBO) on the log marginal likelihood

$$p(y) = \int p(y|f^{(2)}, f^{(1)}, z) \prod_{\ell=1}^2 p\left(f^{(\ell)}\right) p\left(z\right) dz df^{(1)} df^{(2)}, \tag{A.2}$$

by repeatedly applying Jensen's inequality, which yields the variational bound

$$\log p(y) \geq \sum_n \mathop{\mathbb{E}}_{f^{(1)}, f^{(2)}, z_n} \left[\log p\Big(y\Big|f^{(2)}, f^{(1)}, z_n\Big) \frac{\prod_\ell^2 p(f^{(\ell)}) p\left(z_n\right)}{\prod_\ell^2 q(f^{(\ell)}) q\left(z_n\right)}\right], \tag{A.3}$$

where the expectation is taken over the variational distributions $q(f^{(2)})\, q(f^{(1)})$, and $q(z_n)$. However gradient estimates of this objective may have high variance, since evaluating the log expected likelihood requires Monte Carlo sampling over the final DGP layer (which cannot be evaluated analytically).

To obtain a lower-variance gradient estimator, Salimbeni et al. (2019) derive an alternative, partially-collapsed lower bound. To do so, they apply Jensen's inequality to the expectation over $f^{(2)}$ in Equation (A.2) to get

$$\log p(y|f^{(1)}, z) \geq \sum_n L_n(f^{(1)}, z_n) - D_{\mathrm{KL}}(q(f^{(2)}) \parallel p(f^{(2)})), \tag{A.4}$$

where

$$L_n(f^{(1)}, z_n) \stackrel{\text{def}}{=} \mathbb{E}_{f^{(2)}}[\log p(y_n|f^{(2)}, f^{(1)}, z_n)], \tag{A.5}$$

which can be obtained in closed form for a Gaussian likelihood. Applying the exponential function to both sides of Equation (A.4) then yields

$$p(y|f^{(1)}, z) \geq \exp\left(\sum_n L_n(f^{(1)}, z_n) - \mathrm{KL}(q(f^{(2)})||p(f^{(2)}))\right). \tag{A.6}$$

Expressing the joint distribution with $f^{(2)}$ integrated out, we can write

$$p(y) = \mathop{\mathbb{E}}_{f^{(1)}, z}\left[p(y|f^{(1)}, z) \frac{p\left(f^{(1)}\right) p\left(z\right)}{q\left(f^{(1)}\right) q\left(z\right)}\right], \tag{A.7}$$

and substituting in Equation (A.6) and get the lower bound

$$p(y) \geq \mathbb{E}_{f^{(1)}, z} \exp\left(\sum_n L_n(f^{(1)}, z_n) - \mathrm{KL}(q(f^{(2)})||p(f^{(2)}))\right) \frac{p(z)p(f^{(1)})}{q(z)q(f^{(1)})}. \tag{A.8}$$

Applying Jensen's inequality for the expectation over $f^{(1)}$ and taking the average over K samples from z to reduce the variance of the resulting bound, Salimbeni et al. (2019) obtain the partially-collapsed lower bound

$$\begin{aligned}
\log p(y) &\geq \sum_n \mathop{\mathbb{E}}_{f^{(1)}_{1:K}, z_{n,1:K}} \left[\log \frac{1}{K} \sum_{k=1}^K \frac{e^{\mathbb{E}_{f^{(2)}}\left[\log p(y_n|f^{(2)}, f^{(1)}_k, z_{n,k})\right]} p\left(z_{n,k}\right)}{q\left(z_{n,k}\right)}\right] - \sum_\ell^2 D_{\mathrm{KL}}(q(f^{(\ell)}) \parallel p(f^{(\ell)})) \\
&= \sum_n \mathop{\mathbb{E}}_{f^{(1)}_{1:K}, z_{n,1:K}} \left[\log \frac{1}{K} \sum_{k=1}^K \frac{\mathcal{F}(x_n, y_n, f^{(1)}_k, z_{n,k}) p\left(z_{n,k}\right)}{q\left(z_{n,k}\right)}\right] - \sum_\ell^2 D_{\mathrm{KL}}(q(f^{(\ell)}) \parallel p(f^{(\ell)})) \\
&\stackrel{\text{def}}{=} \mathcal{L}_{\text{DGP-IWVI}},
\end{aligned} \tag{A.9}$$

where $\mathcal{F}(x_n, y_n, f^{(1)}_k, z_{n,k}) \stackrel{\text{def}}{=} \exp\left(\mathbb{E}_{q(f^{(2)})}\left[\log p(y_n \mid f^{(2)}, f^{(1)}_k, z_{n,k})\right]\right)$. This is the bound presented in Section 2.2.

B. Importance-Weighted Variational Inference in VAEs

Training a variational auto-encoder (VAE, Kingma & Welling (2013)) with the standard ELBO can lead to generative models with limited generative ability due to the bound being very loose. One was to improve this is to use a different, tighter, ELBO. Of particular relevance to our work, Burda et al. (2016); Mnih & Rezende (2016) propose *importance-weighted* variational inference (IWVI) to obtain tight variational bounds. Namely, they use the variational objective

$$\mathcal{L}_{\text{IWAE}} \stackrel{\text{def}}{=} \mathop{\mathbb{E}}_{z_{n,1:K}} \left[\log \frac{1}{K} \sum_{k=1}^{K} \frac{p_\theta(x_n, z_{n,k})}{q_\phi(z_{n,k} \,|\, x_n)} \right], \tag{B.10}$$

where $z_{n,k} \sim q_\phi(z_n \,|\, x_n)$ and $p_\theta(x_n, z_{n,k})$ represents the generative model. It is worth noting that this bound is as tight as or tighter than the standard VAE ELBO for all K, i.e. $\mathcal{L}_{\text{IWAE}} \geq \mathcal{L}_{\text{VI-VAE}}$. This follows from the fact that the Monte Carlo estimator of the expected importance weight,

$$\hat{Z}_K^{\text{IWAE}} \stackrel{\text{def}}{=} \frac{1}{K} \sum_{k=1}^{K} w_{n,k}, \text{ with } w_{n,k} = \frac{p_\theta(x_n, z_{n,k})}{q_\phi(z_{n,k} \,|\, x_n)}, \tag{B.11}$$

has decreasing variance for increasing K, with a maximum variance at $K = 1$. Moreover, note that for $z_{n,k} \sim q_\phi(z_n \,|\, x_n)$ and $K \to \infty$, $\hat{Z}_K^{\text{IWAE}}$ converges to the true log marginal likelihood $p_\theta(x_n)$ as the number of importance samples increases and the bound becomes tight everywhere.

C. Reparameterization Gradient Estimator of Importance-Weighted VI

We consider the gradient estimator of $\mathcal{L}_{\text{K}}$ with respect to ϕ, the set of parameters governing the variational distribution over the latent variable z, which is given by

$$\nabla_\phi \mathcal{L}_{\text{K}} = \nabla_\phi \sum_n \mathop{\mathbb{E}}_{f^{(1)}_{1:K}, z_{n,1:K}} \left[\log \frac{1}{K} \sum_{k=1}^{K} \frac{\mathcal{F}\left(x_n, y_n, f^{(1)}_k, z_{n,k}\right) p(z_{n,k})}{q(z_{n,k})} \right], \tag{C.12}$$

where $z_{n,k}$ are samples from $q(z)$ and $f^{(1)}_k$ are samples from the variational distribution $q(f^{(1)})$. If z and $f^{(1)}$ are reparameterizable—that is samples from $q(z)$ and $q(f^{(1)})$ can be expressed as

$$z_{n,k} = \tilde{z}(\epsilon^z_k, \phi) = \mu^z(\phi) + \epsilon^z_k \odot \sqrt{\Sigma^z(\phi)} \tag{C.13}$$

and

$$f^{(1)}(x_n, z_{n,k}) = \tilde{f}(\epsilon^{f^{(1)}}_k, x_n, \tilde{z}(\epsilon^z_k, \phi), \psi^{(1)}) = \mu^{f^{(1)}}(x_n, \tilde{z}(\epsilon^z_k, \phi), \psi^{(1)}) + \epsilon^{f^{(1)}}_k \odot \sqrt{\Sigma^{f^{(1)}}(x_n, \tilde{z}(\epsilon^z_k, \phi), \psi^{(1)})}, \tag{C.14}$$

respectively, where $\tilde{z}(\cdot)$ and $\tilde{f}(\cdot)$ are deterministic functions, $\mu^{f^{(1)}}$ and $\Sigma^{f^{(1)}}$ denote the mean and variance of the predictive distribution over $f^{(1)}$, ϕ and $\psi^{(1)}$ are variational parameters of z and $f^{(1)}$ respectively, and $\epsilon^z_k, \epsilon^{f^{(1)}}_k \sim \mathcal{N}(0,1)$ are both independent of ϕ and $\psi^{(1)}$—then the gradient estimator in Equation (C.12) can be expressed as

$$\begin{aligned}
\nabla_\phi \mathcal{L}_{\text{K}} &= \nabla_\phi \sum_n \mathop{\mathbb{E}}_{\epsilon^z_{1:K}, \epsilon^{f^{(1)}}_{1:K}} \left[\log \frac{1}{K} \sum_{k=1}^{K} \frac{\mathcal{F}\left(x_n, y_n, \tilde{f}(\epsilon^{f^{(1)}}_k, \tilde{z}(\epsilon^z_k, \phi), \psi^{(1)})\right) p(\tilde{z}(\epsilon^z_k, \phi))}{q(\tilde{z}(\epsilon^z_k, \phi))} \right] \\
&= \nabla_\phi \sum_n \iint \prod_{k'=1}^{K} p(\epsilon^z_{k'}) p(\epsilon^{f^{(1)}}_{k'}) \log \frac{1}{K} \sum_{k=1}^{K} \frac{\mathcal{F}\left(x_n, y_n, f^{(1)}_k, z_{n,k}\right) p(z_{n,k})}{q(z_{n,k})} d\epsilon^z_{1:K} d\epsilon^{f^{(1)}}_{1:K} \\
&= \sum_n \iint \prod_{k'=1}^{K} p(\epsilon^z_{k'}) p(\epsilon^{f^{(1)}}_{k'}) \nabla_\phi \log \frac{1}{K} \sum_{k=1}^{K} \frac{\mathcal{F}\left(x_n, y_n, f^{(1)}_k, z_{n,k}\right) p(z_{n,k})}{q(z_{n,k})} d\epsilon^z_{1:K} d\epsilon^{f^{(1)}}_{1:K} \\
&= \sum_n \mathop{\mathbb{E}}_{\epsilon^z_{1:K}, \epsilon^{f^{(1)}}_{1:K}} \left[\nabla_\phi \log \frac{1}{K} \sum_{k=1}^{K} \frac{\bar{\mathcal{F}}\left(x_n, y_n, \tilde{f}(\epsilon^{f^{(1)}}_k, \tilde{z}(\epsilon^z_k, \phi), \psi^{(1)})\right) p(\tilde{z}(\epsilon^z_k, \phi))}{q(\tilde{z}(\epsilon^z_k, \phi))} \right] \\
&\qquad \text{with} \quad \epsilon^z_k \sim \mathcal{N}(0,1) \quad \text{and} \quad \epsilon^{f^{(1)}}_k \sim \mathcal{N}(0,1),
\end{aligned} \tag{C.15}$$

where we were able to move the gradient inside the integral, since the variational distributions $q_\phi(z)$ and $q_{\psi^{(1)}}(f^{(1)})$ are reparameterized as described above with the random variables ϵ_k^z and $\epsilon_k^{f^{(1)}}$ being independent of ϕ. Defining the importance weights as

$$w_{n,k}(f^{(1)}, z) \stackrel{\text{def}}{=} \frac{\mathcal{F}\left(x_n, y_n, f_k^{(1)}, z_{n,k}\right) p(z_{n,k})}{q(z_{n,k})}, \tag{C.16}$$

and having established the dependence of $\mathcal{F}(\cdot)$ on x_n, y_n, $f_k^{(1)}$, and $z_{n,k}$, we can now follow Burda et al. (2016) and evaluate the derivative of the logarithm under reparameterization with respect to ϕ, which yields

$$\nabla_\phi \mathcal{L}_{\mathrm{K}} = \sum_n \mathop{\mathbb{E}}_{\epsilon_{1:K}^z, \epsilon_{1:K}^{f^{(1)}}} \left[\sum_{k=1}^K \frac{w_{n,k}(\tilde{f}(\epsilon_k^{f^{(1)}}, \tilde{z}(\epsilon_k^z, \phi), \psi^{(1)}), \tilde{z}(\epsilon_k^z, \phi))}{\sum_{j=1}^K w_{n,j}(\tilde{f}(\epsilon_k^{f^{(1)}}, \tilde{z}(\epsilon_k^z, \phi), \psi^{(1)}), \tilde{z}(\epsilon_k^z, \phi))} \nabla_\phi \log w_{n,k}(\tilde{f}(\epsilon_k^{f^{(1)}}, \tilde{z}(\epsilon_k^z, \phi), \psi^{(1)}), \tilde{z}(\epsilon_k^z, \phi)) \right]. \tag{C.17}$$

Equation (C.17) is the importance-weighted variational inference gradient estimator for a 2-layer latent DGP model, which we note has the same functional form as the IWVAE gradient estimator—with the exception of the additional expectation over the non-final layer random function(s) in $\mathcal{F}\left(x_n, y_n, f_k^{(1)}, z_{n,k}\right)$.

The Monte Carlo gradient estimator is then given by

$$\begin{aligned} \nabla_\phi \mathcal{L}_{\mathrm{K}} = \frac{1}{M} \sum_n \sum_{m=1}^M \sum_{k=1}^K & \frac{w_{n,m,k}(\tilde{f}(\epsilon_{m,k}^{f^{(1)}}, \tilde{z}(\epsilon_{m,k}^z, \phi), \psi^{(1)}), \tilde{z}(\epsilon_{m,k}^z, \phi))}{\sum_{j=1}^K w_{n,m,j}(\tilde{f}(\epsilon_{m,k}^{f^{(1)}}, \tilde{z}(\epsilon_{m,k}^z, \phi), \psi^{(1)}), \tilde{z}(\epsilon_{m,k}^z, \phi))} \\ & \cdot \nabla_\phi \log w_{n,m,k}(\tilde{f}(\epsilon_{m,k}^{f^{(1)}}, \tilde{z}(\epsilon_{m,k}^z, \phi), \psi^{(1)}), \tilde{z}(\epsilon_{m,k}^z, \phi)), \end{aligned} \tag{C.18}$$

where M is the number of Monte Carlo samples used for estimation of $\mathcal{L}_{\mathrm{K}}$.

D. Deep GP Doubly-Reparameterized Gradient Estimator

Assuming that reparameterization of $q(z_n)$ in $\mathcal{L}_{\mathrm{IWAE}}$ is possible, the doubly reparameterized gradient (DREG) estimator of $\mathcal{L}_{\mathrm{IWAE}}$ at a single data point x_n can be expressed as

$$\widetilde{\Delta}_{n,K}^{\mathrm{IWAE}}(\theta, \phi) \stackrel{\text{def}}{=} \nabla_\phi \mathbb{E}_{z_{1:K}} \left[\log \left(\frac{1}{K} \sum_{k=1}^K \bar{w}_{n,k} \right) \right] = \nabla_\phi \mathbb{E}_{\epsilon_{1:K}} \left[\log \left(\frac{1}{K} \sum_{k=1}^K \bar{w}_{n,k} \right) \right] = \mathbb{E}_{\epsilon_{1:K}} \left[\sum_{k=1}^K \frac{\bar{w}_{n,k}}{\sum_j \bar{w}_{n,j}} \nabla_\phi \log \bar{w}_{n,k} \right], \tag{D.19}$$

with $\bar{w}_{n,k} = p_\theta\left(x_n, z_{n,k}\right) / q_\phi\left(z_{n,k} | x_n\right)$. Following Tucker et al. (2019) we can rearrange the $\nabla_\phi \log \bar{w}_{n,k}$ term and take a Monte Carlo estimate of $\epsilon_{1:K}$ to produce the gradient estimate

$$\widetilde{\Delta}_{n,M,K}^{\mathrm{IWAE}}(\theta, \phi) \stackrel{\text{def}}{=} \frac{1}{M} \sum_{m=1}^M \sum_{k=1}^K \frac{\bar{w}_{n,m,k}}{\sum_{k'=1}^K \bar{w}_{n,m,k'}} \left(\frac{\partial \log \bar{w}_{n,m,k}}{\partial z_{n,m,k}} \frac{\partial z_{n,m,k}}{\partial \phi} - \frac{\partial}{\partial \phi} \log q_\phi(z_{n,m,k} \,|\, x_n) \right), \tag{D.20}$$

where $z_{n,m,k} \overset{\text{i.i.d.}}{\sim} q_\phi(z_n \,|\, x_n)$ as before. As stated in Tucker et al. (2019), the SNR of this gradient estimator scales as $\mathcal{O}(\sqrt{K})$ instead of $\mathcal{O}(1/\sqrt{K})$, that is, the SNR improves as K increases.

To derive a DREG estimator for DGPs, we note that in Appendix C, we show that the reparameterization gradient for the gradient with respect to the parameters of the variational distribution over the latent variable z in IWVI in DGPs (see Equation (3)) at a single data point x_n can be expressed as

$$\Delta_{n,M,K}^{\mathrm{DGP}}(\phi) \stackrel{\text{def}}{=} \frac{1}{M} \sum_{m=1}^M \sum_{k=1}^K \frac{w_{n,m,k}}{\sum_{j=1}^K w_{n,m,j}} \nabla_\phi \log w_{n,m,k}, \tag{D.21}$$

$$\text{where} \quad w_{n,m,k} \stackrel{\text{def}}{=} \tilde{\mathcal{F}}_{n,m,k} \frac{p(\tilde{z}(\epsilon_{m,k}^z, \phi))}{q(\tilde{z}(\epsilon_{m,k}^z, \phi))} \quad \text{and} \quad \tilde{\mathcal{F}}_{n,m,k} \stackrel{\text{def}}{=} \bar{\mathcal{F}}(x_n, y_n, \tilde{f}(\epsilon_{m,k}^{f^{(1)}}, \tilde{z}(\epsilon_{m,k}^z, \phi), \psi^{(1)})).$$

Evaluating the $\nabla_\phi \log w_{n,m,k}$ term as above, we get

$$\Delta^{\text{DGP}}_{n,M,K}(\phi) \overset{\text{def}}{=} \frac{1}{M}\sum_{m=1}^{M}\sum_{k=1}^{K}\frac{w_{n,m,k}}{\sum_{j=1}^{K} w_{n,m,j}}\left(\frac{\partial \log w_{n,m,k}}{\partial \tilde{z}}\frac{\partial \tilde{z}(\epsilon^z_{m,k})}{\partial \phi} - \frac{\partial}{\partial \phi}\log q(\tilde{z}(\epsilon^z_{m,k},\phi))\right), \tag{D.22}$$

and note that this estimator strongly resembles the IWAE-DREG estimator with the difference that here, $\frac{\partial \log w_{n,m,k}}{\partial \tilde{z}}$ contains the reparameterized DGP layers. We exploit this similarity to derive a DREG estimator for DGPs by following the derivation presented in Tucker et al. (2019). To do so, we take advantage of the fact that once the reparameterization gradient estimator is established, the derivation in Tucker et al. (2019) only relies on the equivalence between the REINFORCE (Williams, 1992) and reparameterization gradient estimators and does not rely on the exact expression for $\frac{\partial \log w_{n,m,k}}{\partial \tilde{z}}$. This way, we are able to replace the high-variance REINFORCE gradient term, $\frac{\partial}{\partial \phi}\log q(\tilde{z}(\epsilon^z_{m,k},\phi))$ in Equation (D.21) and, obtain a DREG estimator for IWVI for DGPs given by

$$\widetilde{\Delta}^{\text{DGP}}_{n,M,K}(\phi) \overset{\text{def}}{=} \frac{1}{M}\sum_{m=1}^{M}\sum_{k=1}^{K}\left(\frac{w_{n,m,k}}{\sum_{j=1}^{K} w_{n,m,j}}\right)^2\frac{\partial \log w_{n,m,k}}{\partial \tilde{z}}\frac{\partial \tilde{z}(\epsilon^z_{m,k})}{\partial \phi}, \tag{D.23}$$

where, unlike for IWAEs,

$$w_{n,m,k} \overset{\text{def}}{=} \tilde{\mathcal{F}}_{n,m,k}\frac{p(\tilde{z}(\epsilon^z_{m,k},\phi))}{q(\tilde{z}(\epsilon^z_{m,k},\phi))} \quad \text{with} \quad \tilde{\mathcal{F}}_{n,m,k} \overset{\text{def}}{=} \bar{\mathcal{F}}(x_n, y_n, \tilde{f}(\epsilon^{f^{(1)}}_{m,k}, \tilde{z}(\epsilon^z_{m,k},\phi), \psi^{(1)})). \tag{D.24}$$

E. Experimental Details

E.1. Datasets

We design a 1D "demo" dataset which exhibits multimodality and is easy to visualize. It is defined

$$f_1(x) = \frac{\sin(4x)}{3} + 0.2e^{\epsilon} \tag{E.25}$$

$$f_2(x) = \frac{9x^2}{30} + 1.5 + 0.2e^{\epsilon} \tag{E.26}$$

$$\epsilon \sim \mathcal{N}(0,1) \tag{E.27}$$

$$f(x) = \begin{cases} f_1(x), & \text{with probability } 0.6 \\ f_2(x), & \text{otherwise.} \end{cases} \tag{E.28}$$

Figure 7 shows the demo dataset. It contains 2000 points.

In Figure 1 and Figure 2 we consider the demo dataset.

In Figure 3 and Figure 4 we consider a set of real world datasets. These are the UCI datasets (Dua & Graff, 2017). We reserve 10% of the dataset as test data, and use the remaining data as training data. We normalize the data so that the training set has mean zero and standard deviation one. We use the following library to load and preprocess the data, and it gives full details of what steps are performed:

https://github.com/hughsalimbeni/bayesian_benchmarks.

E.2. Models, Optimization, and Initialization

We use the same configuration and hyperparameters in all experiments, with the exception of the number of DGP layers, number of importance samples, batch size, and number of training iterations. We give the configuration of the other hyperparameters below.

Latent Variable All models considered in the empirical evaluation use latent variables concatenated to the input data.

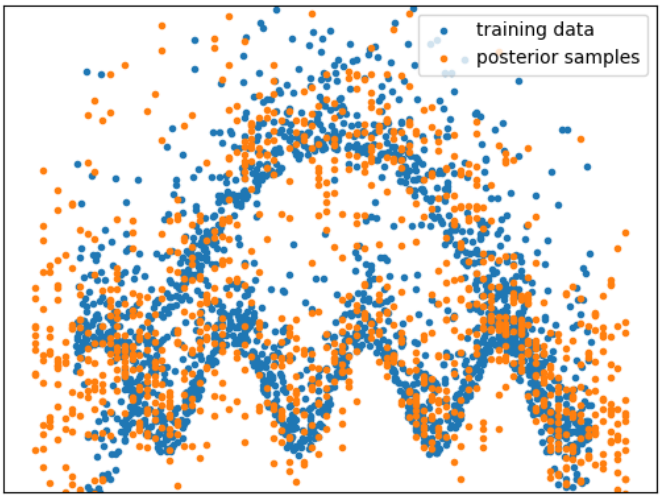


Figure 7. 1D demo dataset with multimodality. The blue points show the training data, while the orange points are samples from the posterior distribution of a two-layer DGP fit to the data.

Parameterization of $q(z)$ We use a fully connected neural network with two hidden layers of 20 units each, skip connections between each layer, and the $\tanh(\cdot)$ non-linearity. The network has two heads, for the mean and standard deviation of $q(z)$ respectively. Following Salimbeni et al. (2019), we add a bias of -3 to the standard deviation head to ensure that the output is small at initialization, and apply the softplus function to enforce that the output is positive. We initialize the weights using the scheme introduced by Glorot & Bengio (2010).

The other hyperparameters and initialization schemes match those given by Salimbeni et al. (2019), which we reproduce below for completeness:

Kernels RBF with ARD. Lengthscale initialized to the square root of the dimension.

Inducing points 128 points per DGP layer. Initialized to the data, if 128 data points or fewer. Otherwise initialized to the cluster centroids returned by kmeans2 from the SciPy package, with 128 points. The first DGP layer has an additional input dimension due to the latent variable. The inducing points for this dimension are initialized to random draws from a standard normal distribution.

Linear projections between layers We implement the linear mean functions and multioutput structure using a linear projection of five independent GPs concatenated with their inputs. We initialized the projection matrix to the first five principle components of the data concatenated with the identity matrix.

Likelihood We initialize the likelihood variance to 0.01.

Parameterizations All positive model parameters are constrained to be positive using the softplus function, clipped to 10^{-6}. The variational parameters for the sparse GP layers are parameterized by the mean and the square root of the covariance.

Optimization For the final DGP layer we use natural gradients on the natural parameters, with an initial step size of 0.01. All other parameters are optimized using the Adam optimizer (Kingma & Ba, 2015) with all parameters set to their TensorFlow default values except the initial step size of 0.005. We anneal the learning rate of both the Adam and natural gradient steps by a factor of 0.98 per 1000 iterations. The mean functions and kernel projection matrices (the P matrices that correlate the outputs) are not optimized.

E.3. Details by Figure

Figure 1

1. Train a 2-layer model to convergence of the ELBO on the demo dataset.
2. For a single point randomly chosen from the training set, take $Q = 1000$ samples of the gradient estimator. This yields a $Q \times P$ tensor, where P is the number of parameters of the variational distribution over the latent variable, $q(z)$.
3. Compute the SNR, by taking the mean and standard deviation over the Q dimension.
4. Plot the SNR for 4 randomly chosen parameters.

Figure 2 Train a 2-layer model to convergence of the ELBO on the demo dataset. Randomly choose a single data point from the training set and a single parameter of $q(z)$. Sample gradient estimates for this parameter at the data point.

Figure 3

1. Train models of depths $1, 2, 3, 4$ to convergence of the ELBO.
2. For 10 randomly chosen data points in the training set:
   (a) Take $Q = 1000$ samples of the gradient estimator yielding a $Q \times P$ tensor, where P is the number of parameters of the variational distribution over the latent variable, $q(z)$.
   (b) Compute the SNR, by taking the mean and standard deviation over the Q dimension.
   (c) Compute the mean SNR, over the P dimension.
3. Take the mean of the SNR over the 10 points.

Figure 4 and Table 1 We train for $300,000$ iterations.

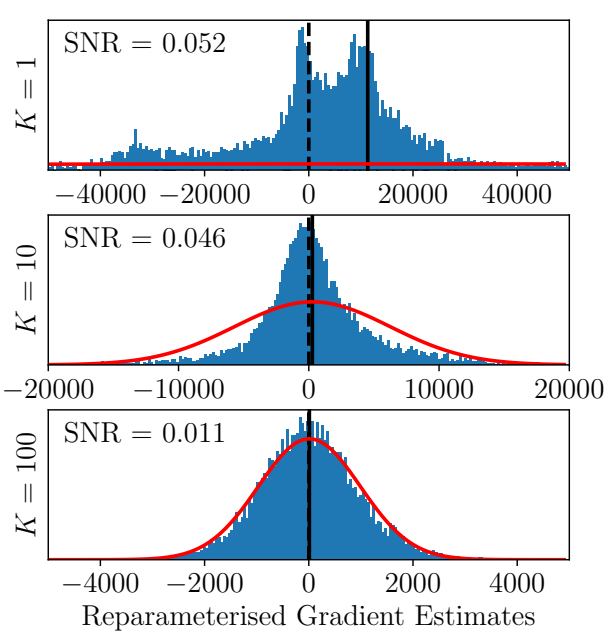
We estimate the test log-likelihood using Monte Carlo sampling:

1. For each point $(x_n y_n)$ in the test set:
   (a) Take 10000 samples of $z_{n,k} \sim \mathcal{N}(z \,|\, 0, I)$, sample $f_{n,k} \sim f(x_n, z_{n,k})$ for each $z_{n,k}$, evaluate the marginal likelihood $\mathcal{N}(y_n \,|\, f_{n,k}, \sigma^2)$
   (b) Take the mean of the sampled marginal likelihood values
2. Take the mean of the marginal likelihood over all data points

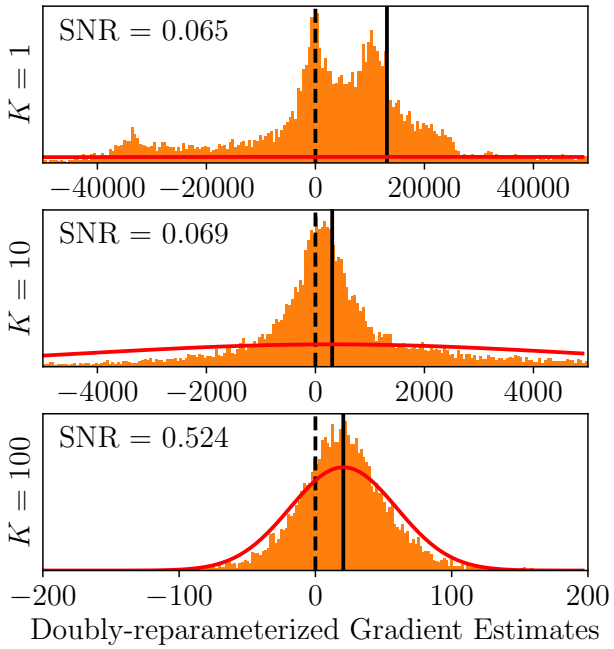
We find that some train/test splits had significantly worse performance, so we exclude them as outliers. The criteria we use for this is a test log-likelihood more than two standard deviations from the mean for that dataset and model configuration. We are careful to exclude the results for both REG and DREG, or $q(z)$ and $p(z)$, for a particular split and dataset, even if just one of them was an outlier.

Figure 5 We take the relevant trained models from Figure 4. We evaluate each 10000 times on one randomly selected point from the test set, and plot the resulting y values.

F. Additional results



(a) Histogram of REG estimates.



(b) Histogram of DREG estimates.

Figure 8. Histograms of REG (a) and DREG (b) estimates for varying numbers of importance samples, K, with $K = 1, 10, 100$. The solid vertical lines denote the mean of the empirical distributions of gradient estimates, and the dashed vertical lines mark zero. The red curves denote Gaussian probability densities with the same mean and standard deviation as the empirical distributions. Note that for $K = 1$ there are a number of very large outliers which we have truncated from the histogram.

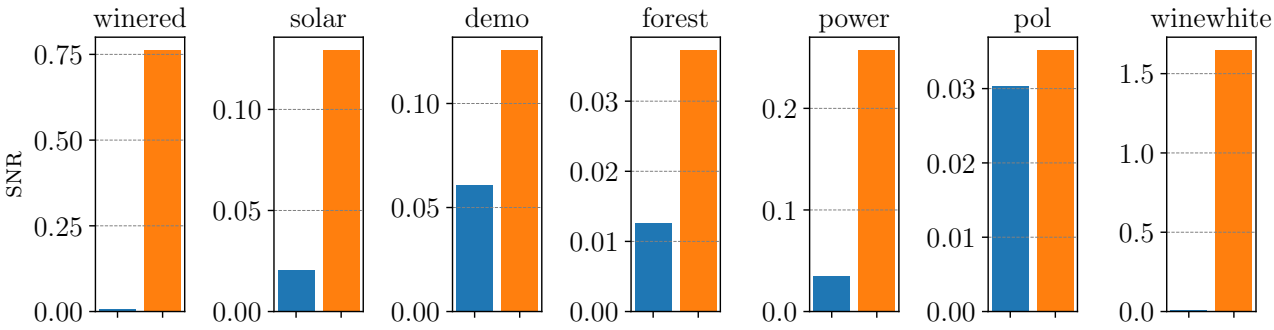


Figure 9. SNR of the REG (left, blue) and DREG (right, orange) estimates in the configuration used for Table 1, confirming that the DREG estimates have a substantially better SNR. The shaded areas show the interquartile range over two seeds.