Flexible Approximate Inference via Stratified Normalizing Flows

Chris Cundy
Computer Science Department
Stanford University
cundy@stanford.edu

Stefano Ermon
Computer Science Department
Stanford University
ermon@stanford.edu

Abstract

A major obstacle to forming posterior distributions in machine learning is the difficulty of evaluating partition functions. Monte-Carlo approaches are unbiased, but can suffer from high variance. Variational methods are biased, but tend to have lower variance. We develop an approximate inference procedure that allows explicit control of the bias/variance tradeoff, interpolating between the sampling and the variational regime. We use a normalizing flow to map the integrand onto a uniform distribution. We then randomly sample regions from a partition of this uniform distribution and fit simpler, local variational approximations in the image of these regions through the flow. When a partition with only one region is used, we recover standard variational inference, and in the limit of an infinitely fine partition we recover Monte-Carlo sampling. We show experiments validating the effectiveness of our approach.

1 INTRODUCTION

Integration of high-dimensional functions is a central problem in many fields, serving as the workhorse that powers posterior inference in probabilistic machine learning, inference with latent-variable models, and risk-sensitive modelling in finance (Dick et al., 2013). However, methods that work well in low dimensions can quickly become computationally intractable as the dimension increases: this is the so-called *curse of dimensionality*. This challenge motivates the development of approximate methods. The two main families of approximate methods are: (1) variational approaches, which fit an approximating function to the integrand, and (2) Monte-Carlo methods, which take random samples from the domain of the integrand to compute its average value (Bishop, 2006). While Monte-Carlo methods are guaranteed to give the correct result in the limit of infinite samples (i.e., they are unbiased), they can suffer from high variance. Conversely, variational methods are biased, but the resulting Evidence Lower Bound (ELBO) (Ranganath et al., 2013) has zero variance if evaluated analytically. Monte Carlo methods for approximating the ELBO introduce some variance, but this variance is typically smaller than pure sampling-based methods. Viewing the two families of methods as two extremes on the bias-variance tradeoff (Bishop, 2006), we observe that it is frequently useful to move along the bias/variance curve, e.g. incurring some bias in order to reduce the variance to manageable levels, or in order to speed up computation time.

We describe a method that allows continuous interpolation along the bias-variance curve, from the high-bias, low variance variational inference (VI) regime to the high-variance, low-bias sampling regime. We randomly sample from a partition of the integration domain and fit variational approximations inside each ‘cell’ of the partition. Stratification in Monte-Carlo methods refers to sub-sampling the integration domain to increase sample-efficiency: we can view our method as a form of stratification for variational inference. In the limiting case where the size of the cells goes to zero, we recover ordinary importance sampling, where the proposal distribution is given by the normalizing flow’s induced probability distribution, and in the limit with a single partition cell we recover ordinary variational inference.

The variance of our method depends heavily on the choice of partition. Similarly to importance sampling (IS), we want to choose a partition which is adapted to the objective function, with most of the partition cells in regions where the objective function is large. Normalizing flows provide a natural method to provide this partition of the integration domain, by choosing a uniform partition of the flow’s base space and observing the image in the integration domain. In a toy experiment we show that the

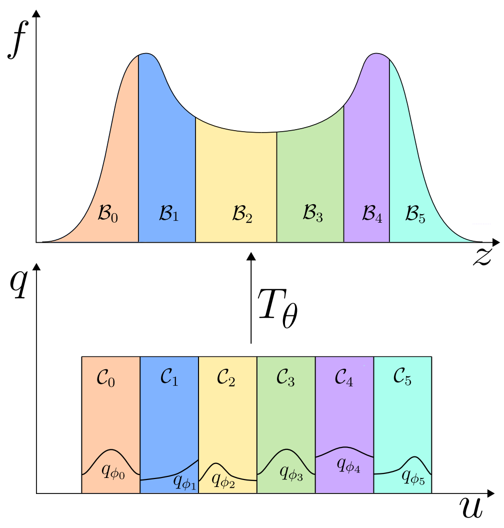

Figure 1: A schematic illustrating our approach. Regular sub-regions of a uniform density are mapped via an invertible transformation T_θ (learned via a normalizing flow) onto regions of $\mathcal{Z}$ with equal f-density. We then randomly sample a cell $\mathcal{C}_i$ in $\mathcal{U}$, and fit a variational density q_{ϕ_i} which via T_θ induces a distribution on $\mathcal{B}_i$. We compute the corresponding ELBO and repeatedly sample further cells, combining the ELBO estimates to obtain an estimate of $\int_{\mathcal{Z}} f(z)dz$. We do not need to exhaustively sample all the cells, some random subset is sufficient. The number of sampled cells and size of the cells determines the bias/variance tradeoff.

partitioning scheme gives lower ELBOs than the same architecture fitted variationally. In a Bayesian linear regression experiment where we show that our method achieves the same result as exhaustive sampling with hundreds of millions of points, while the ordinary variational approach suffers from high bias.

2 BACKGROUND

Many problems require high-dimensional integration. An example is computing the posterior distribution of a parameter θ over its possible values Θ given some data D and a likelihood function $P(D|\theta)$: from Bayes' theorem $P(\theta|D) = P(D|\theta)P(\theta)/P(D)$. Since $P(D) = \int_\Theta P(D|\theta')P(\theta')d\theta'$, to compute $P(D)$ we must integrate over a possibly very high-dimensional space if the dimensionality of θ is high. Similarly, in latent variable models we must marginalise out latent variables h with possible values H, so that e.g. $P(y|x) = \int_H P(y|x,h)P(h)dh$. More generically, we can pose the goal of an approximate inference algorithm: to provide an estimate of

$$\int_{\mathcal{Z}} f(z)dz, \tag{1}$$

with $\mathcal{Z} \subset \mathbb{R}^d$ and $f \geq 0$.

In low dimensions, it is feasible to compute integrals to high accuracy by evaluating f on a grid of points, but the number of points needed grows exponentially with the dimension d (Papageorgiou & Wasilkowski, 1990). Monte-Carlo methods forego exhaustive enumeration on a regular grid in favour of randomly choosing points in $\mathcal{Z}$ and forming the empirical average. If the N points are drawn from a known distribution q with $q(z) > 0$ for points where $f(z) > 0$, we find

$$\int_{\mathcal{Z}} f(z)dz = \mathbb{E}_{z\sim q}\left[\frac{f(z)}{q(z)}\right] \approx \frac{1}{N}\sum_{i=1}^{N}\frac{f(z_i)}{q(z_i)}, \tag{2}$$

where $z_i \sim q$. When the distribution q is proportional to f, this estimator has zero variance, and will generally have lower variance if q has high density in regions where f has high values (Owen & Zhou, 2000).

In contrast to Monte-Carlo methods, variational methods maximize a lower bound to $\int_{\mathcal{Z}} f(z)dz$, noting from Jensen's inequality that for any distribution q,

$$\log \mathbb{E}_{z\sim q}\left[\frac{f(z)}{q(z)}\right] \geq \mathbb{E}_{z\sim q}\left[\log f(z) - \log q(z)\right]. \tag{3}$$

The quantity on the right hand side is known as the evidence lower bound (ELBO). By parameterizing the distribution q with a parameter θ from a set Θ and maximizing the ELBO with respect to θ, we can obtain a lower bound on the value of the integral. In the extreme case when $f \propto q$, the lower bound becomes tight. For each variational family corresponding to a set of possible parameters Θ, there is a set of parameters θ which achieve the highest ELBO. Assuming we can carry out the optimization to find such a θ, and q and f are sufficiently simple so that equation 3 can be evaluated analytically (e.g., mean field inference in graphical models), this approximation is deterministic. Even when equation 3 is estimated via Monte Carlo as in black box variational inference (Ranganath et al., 2013), the variance of the ELBO estimate is typically small.

This characterizes the *bias-variance tradeoff* involved in the choice between variational inference and importance sampling. Variational inference will always return an underestimate of the objective, but typically has low variance. The importance sampling approach is unbiased but can have very high variance, so may take an unfeasibly large amount of time to recover an accurate estimate. With a poor proposal distribution, the estimate may even appear to converge quickly, but to an incorrect value. This is because significant contributions to the estimate can come from very rare points under the proposal, with large importance weights (Owen & Zhou, 2000). Our method allows interpolation between the fully-variational and the fully-importance-sampled extremes, allowing the amount of bias and variance to be tuned according to the specific problem.

2.1 NORMALIZING FLOWS

A normalizing flow (Rezende & Mohamed, 2015; Papamakarios et al., 2019) is a density model constructed by applying an invertible transformation $T : \mathcal{U} \to \mathcal{Z}$ to a base density π_0, resulting in a density q on $\mathcal{Z}$. The change of variables equation gives an expression for q:

$$q(T(u)) = \pi_0(u) \left| \det \frac{dT}{du} \right|, \tag{4}$$

where $\frac{dT}{du}$ is the Jacobian of T with respect to u. Flow models allow for exact likelihood evaluation, unlike some other models, such as variational autoencoders (Kingma & Welling, 2013), where we can only tractably evaluate a lower bound on the likelihood. However, the exact likelihood comes at a cost: parameterising T such that it is invertible is not straightforward, and can require a large number of parameters to develop expressive flow models.

Most density-modelling applications of flows involve modelling the inverse transform T^{-1} Dinh et al. (2017); Grathwohl et al. (2019), using the equivalent formulation $q(z) = \pi_0(T^{-1}(z)) \left| \det \frac{dT^{-1}}{dz} \right|$, since learning a flow to maximize the likelihood of samples z from a distribution requires evaluating and optimizing $q(z)$. By contrast, our application does not assume that we can sample from our objective f (in fact f does not even need to be a normalized density). Therefore, we train our flow by taking samples u from the base distribution, and only model the forward transform T. Since we implemented our flow using RealNVP, a coupling flow, we can tractably evaluate both T and T^{-1}. However these considerations are important if future work uses autoregressive flows, where generally only one of T or T^{-1} are able to be tractably evaluated, and computing the other may be up to d times slower, for a d-dimensional flow (Papamakarios et al., 2019).

The base distribution π_0 is generally chosen to be a unit multivariate normal distribution, which has a tractable density, unbounded support, and is easy to sample from. However, we use a base distribution which is the uniform distribution on the unit cube $[0, 1]^d$, since we want to choose partition cells with equal probability mass. These are easily obtained with cubes in the base space for the uniform distribution, and is somewhat more involved in the Gaussian case. Using the uniform density requires some care, which we discuss in section 4.2.

3 METHOD

Recall that our goal is to compute integrals of the form $\int_{\mathcal{Z}} f(z)dz$. Given a distribution q, we have a lower bound:

$$\log \int_{\mathcal{Z}} f(z) \geq \mathbb{E}_{z\sim q} \left[\log f(z) - \log q(z)\right]. \tag{5}$$

We can form a finite partition P_N of $\mathcal{Z}$: a family of N disjoint nonempty sets $\mathcal{B}_i$ such that $\bigcup_i \mathcal{B}_i = \mathcal{Z}$. In each cell we define a separate distribution q_i (supported only on $\mathcal{B}_i$) and observe that

$$\int_{\mathcal{B}_i} f(z)dz \geq \exp\left(\mathbb{E}_{z\sim q_i} \left[\log f(z) - \log q_i(z)\right]\right). \tag{6}$$

Summing the lower bounds from each cell, we have

$$\int_{\mathcal{Z}} f(z)dz \geq \sum_i \exp \mathbb{E}_{z\sim q_i} \left[\log \frac{f(z)}{q_i(z)}\right]. \tag{7}$$

If we introduce a distribution Q_N over cells in P_N, we can introduce importance-weighting to obtain our lower bound $\mathcal{L}_{Q_N}$,

$$\begin{aligned}\int_{\mathcal{Z}} f(z)dz &\geq \mathbb{E}_{\mathcal{B}_i\sim Q_N} \left[\frac{1}{Q_N(\mathcal{B}_i)} \exp \mathbb{E}_{z\sim q_i} \left[\log \frac{f(z)}{q_i(z)}\right]\right] \\ &= \mathcal{L}_{Q_N}.\end{aligned} \tag{8}$$

This expression reduces to familiar cases in two natural limits. In the case with a single cell which encompasses $\mathcal{Z}$, we have the standard variational lower bound. At the other extreme, in the limit when the size of the cells is very small, f is approximately constant, and so the q_i that gives the tightest bound is the uniform distribution over $\mathcal{B}_i$. The family of distributions Q_N over cells approaches a distribution Q over points in $\mathcal{Z}$, and so we recover a lower bound $\mathbb{E}_{z\sim Q}\left[f(z)/Q(z)\right]$, which is the standard importance sampling expression. A special case that we will use heavily is when we choose uniformly from a partition $\mathcal{R}$ splitting $\mathcal{Z}$ into N pieces, where we get a lower bound

$$\mathcal{L}_N = N\mathbb{E}_{\mathcal{B}_i\sim U(\mathcal{R})} \left[\exp \mathbb{E}_{z\sim q_i} \left[\log f(z) - \log q_i(z)\right]\right], \tag{9}$$

which we can estimate with an estimator

$$\hat{\mathcal{L}}_N = \frac{N}{n} \sum_{i=1}^{n} \exp\left(\frac{1}{n'} \sum_{j=1}^{n'} \log f(z_{i,j}) - \log q_i(z_{i,j})\right), \tag{10}$$

where we take the average from a sample of n cells (out of a partition with N cells), in each cell computing the ELBO using n' samples, where $z_{i,j}$ is the jth point in a batch of n' points drawn from the distribution q_i.

Although $\mathcal{L}_N$ is always a valid lower bound, we want the tightest lower bound possible. The key idea behind our approach is that since $\mathcal{L}_N$ is valid for all choices of inner-cell distributions q_i, we can improve the estimator by optimizing each q_i before evaluating $\hat{\mathcal{L}}_N$. Since we expect the variational optimization task in each cell to be easier than optimization over the entire space, we would hope to be able to find a q_i which is a close approximation to f. However, the easier optimization task in each cell comes at the cost of increased variance in the estimate of the objective $\mathcal{L}_N$ as we try to estimate the global ELBO with individual ELBOs. The pseudocode in algorithm 1 describes this procedure. We can also interpret our approach as lazily sampling parts of a very good variational function, which is defined piecewise in the cells of $\mathcal{Z}$.

Theorem 1 illustrates that this procedure reduces to estimation of $\int_{\mathcal{Z}} f(z)dz$ using importance sampling in the limit of $N \to \infty$, and Theorem 3 (in the appendix) shows that finite-sample estimates of $\mathcal{L}_N$ are indeed lower bounds with high probability. In the next section, we discuss how to find a good partition of $\mathcal{Z}$, which determines the variance of our estimator.

Theorem 1. *For any distribution q, we can construct a sequence of partitions P_N and probability distributions over the cells Q_N such that the distribution of points obtained by sampling a cell from Q_N then uniformly sampling from that cell converges in distribution to q. Under that choice of P_N and Q_N, the importance-weighted exponential of the ELBO obtained in the cell as a function of z converges pointwise to $f(z)/q(z)$.*

Proof. We choose a partition of $\mathbb{R}^d$ into rectangular cells with side length 2^{-n}, and a probability distribution over the cells $Q_N(\mathcal{B}_i) = \int_{\mathcal{B}_i} q(z')dz'$. If we choose a point Z in $\mathcal{Z}$ by choosing a cell $\mathcal{B}_i$ with probability $Q_N(\mathcal{B}_i)$ and then choose a point uniformly in the cell, Z has a piecewise-constant density $R_N(z) = \frac{1}{\mathrm{Vol}(\mathcal{B}(z))} \int_{\mathcal{B}(z)} q(z')dz'$, where we define $\mathcal{B}(z) = \{\mathcal{B}_i : z \in \mathcal{B}_i\}$. By the Lebesgue differentiation theorem, $\lim_{N\to\infty} R_N(z) = q(z)$ almost everywhere, and since the densities are a.e. the same, the random variables converge in distribution.

Now consider as a function of z, the corresponding value obtained, $S_N(z) = Q_N(\mathcal{B}(z))^{-1} \exp \mathrm{ELBO}(\mathcal{B}(z))$. In the limit $n \to \infty$, we have

$$\lim_{N\to\infty} S_N(z) = \lim_{N\to\infty} \frac{1}{Q_N(\mathcal{B}(z))} \exp\left[\mathop{\mathbb{E}}_{z'\sim q_i} \log \frac{f(z')}{q_i(z')}\right]. \tag{11}$$

Noting that our choice of the uniform distribution over the cell has $q_i(z) = \mathrm{vol}(\mathcal{B}_i(z))$, we also have $Q_N(\mathcal{B}_i) = R_N(z)\,\mathrm{vol}(\mathcal{B}_i)$, so cancelling the $\log q$ term we obtain

$$\lim_{N\to\infty} S_N(z) = \lim_{N\to\infty} \frac{1}{R_N(z)} \exp \int_{\mathcal{B}(z)} \frac{\log f(z')}{\mathrm{vol}(\mathcal{B}(z))} dz', \tag{12}$$

so $\lim_{N\to\infty} S_N(z) = f(z)/q(z)$. □

Theorem 2. *For a variational distribution q supported on $\mathcal{Z}$ with ELBO $\mathrm{ELBO}(\mathcal{Z})$, and a partition of $\mathcal{Z}$ into two parts $\mathcal{A}, \mathcal{B}$ with $\int_{\mathcal{A}} q = \int_{\mathcal{B}} q$, $\log\left[\exp \mathrm{ELBO}(\mathcal{A}) + \exp \mathrm{ELBO}(\mathcal{B})\right] \geq \mathrm{ELBO}(Z)$, where $\mathrm{ELBO}(A)$ is the renormalized q with $q(z) = 0$ if $z \notin \mathcal{A}$, $q(z)/\int_{\mathcal{A}} q(z')dz'$ otherwise, similarly for $\mathcal{B}$.*

Proof. Expanding $\mathrm{ELBO}(\mathcal{Z})$,

$$\mathrm{ELBO}(\mathcal{Z}) = \tfrac{1}{2} \int_{\mathcal{Z}} q_{\mathcal{A}}(z) \left[\log \frac{f(z)}{q_{\mathcal{A}}(z)} + \log 2\right] dz \tag{13}$$

$$+ \tfrac{1}{2} \int_{\mathcal{Z}} q_{\mathcal{B}}(z) \left[\log \frac{f(z)}{q_{\mathcal{B}}(z)} + \log 2\right] dz \tag{14}$$

$$= \tfrac{1}{2} \mathrm{ELBO}(\mathcal{A}) + \tfrac{1}{2} \mathrm{ELBO}(\mathcal{B}) + \log 2, \tag{15}$$

If we use those two ELBO estimates separately in our estimator and compare to the log of our lower bound,

$$\log \mathcal{L}_2 = \log\left[\exp \mathrm{ELBO}(\mathcal{A}) + \exp \mathrm{ELBO}(\mathcal{B})\right],$$

using Jensen's inequality,

$$\log \mathcal{L}_2 \geq \frac{1}{2}\left[\mathrm{ELBO}(\mathcal{A}) + \mathrm{ELBO}(\mathcal{B}) + 2\log 2\right] \tag{16}$$

$$\geq \mathrm{ELBO}(\mathcal{Z}). \tag{17}$$

□

4 IMPROVING OUR ESTIMATOR

Recall our estimator from equation (10). We can see that its bias and variance will depend on both our choice of a partition of $\mathcal{Z}$ and of the individual variational distributions q_i. In this section we discuss how to find a good partition which will allow us to train flexible variational distributions.

4.1 CHOICE OF PARTITION

We can see from inspection of equation (10) that the variance of $\hat{\mathcal{L}}_N$ is minimized when the ELBO is the same

```
Algorithm 1: Pseudocode for Evaluating $\hat{\mathcal{L}}_{Q_N}$
Input : Set of cells $P$ partitioning $\mathcal{Z}$ into $N$ parts,
        distribution $Q$ assigning probability $Q(\mathcal{B}_i)$ to
        cell $\mathcal{B}_i$, number of cells to sample $n$, number of
        inner optimization steps $m$, objective function $f$.
output: Estimate of $\hat{\mathcal{L}}_{Q_N}$, with $\int_{\mathcal{Z}} f dz \geq \hat{\mathcal{L}}_{Q_N}$ with
        high probability given sufficiently large $n$.
for $i \in (1, \ldots, n)$ do
    Initialize distribution $q_{\theta i}$ with parameters $\theta_i$ from
    random
    Sample $\mathcal{B}_i$ from $Q_N$
    for $j \in (1, \ldots, m)$ do
        Sample $n'$ points from $\mathcal{B}_i$
        Compute $\text{ELBO}(\mathcal{B}_i) = \frac{1}{n'} \sum_{n'} \log \frac{f(z)}{q_i(z)}$
        Step $\theta_i$ in direction of $\nabla_{\theta i} \text{ELBO}(\mathcal{B}_i)$
    end
    Store $S_i = Q_N(\mathcal{B}_i)^{-1} \exp \text{ELBO}(\mathcal{B}_i)$
end
Return $\frac{N}{n} \sum_i S_i$
```

across all partition cells. We do not know what the ELBO in each cell will be, as it depends on the specifics of the optimization procedure and variational family. However, we know that $\text{ELBO}(\mathcal{B}_i) \leq \int_{\mathcal{B}_i} f(z)dz$, and so a useful heuristic would be to choose a partition such that that the total density of f in each cell is equal. However, in general such a partitioning would be difficult to specify explicitly for a complicated f, as each cell would have a complicated shape. Therefore, we can leverage the properties of normalizing flows to obtain this partitioning.

If we train a normalizing flow to have density proportional to f, equal-probability-mass regions in the base space will be mapped to equal f-mass regions in $\mathcal{Z}$. The simplest approach, and the one we will take here, is to learn a flow with its base density supported uniformly on the unit cube $(0, 1)^d$. We can then choose a partition of the unit cubes into smaller cubes. The image of the smaller cubes under the flow transformation are the cells in the partition of $\mathcal{Z}$. We learn the flow by minimizing the KL-divergence

$$D_{\text{KL}}(q(z) \| \tilde{f}(z)) = \mathop{\mathbb{E}}_{z \sim q} \left[\log \frac{q(z)}{f(z)} - \log \int_{\mathcal{Z}} f(z')dz' \right], \tag{18}$$

where $\tilde{f}(z) = f(z) / \int_{\mathcal{Z}} f(z)dz$, so that $\tilde{f}$ is a probability distribution. However, the normalization term does not appear during optimization of q, since it is independent of q. Note that we cannot use the forward KL-divergence because we do not assume that we can sample from $\hat{f}$. Finally, we must ensure that our partition covers the entirety of $\mathcal{Z}$ by designing q such that it has support everywhere in $\mathcal{Z}$. This condition is generally satisfied for most flow architectures if we use a base distribution with unbounded support such as the normal, but it is not necessarily satisfied for our uniform distribution. We discuss how we ensure the flow has unbounded support in section 4.4.

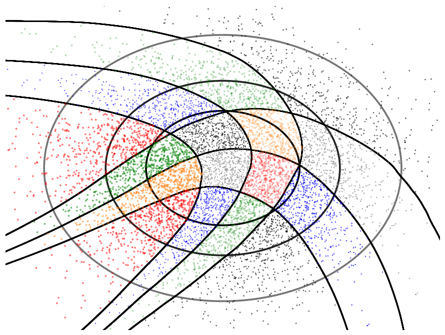
Figure 2: A uniform partition under the image of a flow model trained to fit a 2d unit Gaussian. Uniform cells in the base space are mapped to nonlinear regions in $\mathcal{Z}$. Faint lines show level sets of the Gaussian density, thick lines show grid lines in the base space. Points are uniformly sampled in the base space.

Using a flow to provide our partition has two desirable features. We have a method to learn partitions that will result in lower-variance estimates. Secondly, we have an explicit transformation which maps a regularly-shaped domain (a box in the base space) to a region with a complicated shape. This allows us to perform optimization (e.g. fitting a variational approximation) in the region by fitting in the regularly-shaped cell in the base space and projecting into $\mathcal{Z}$. Figure 2 shows an example of a partitioning from the unit cube to the unit Gaussian.

4.2 OPTIMIZATION OF NORMALIZING FLOW

For many normalizing flow architectures it is important to specify whether the forward transform T or the reverse transform T^{-1} is the primary direction to model. Inspecting the optimization procedure, we see that we have to compute

$$\nabla_\theta D_{\text{KL}} \left(q_\theta(z) \| \tilde{f}(z) \right) = \nabla_\theta \mathbb{E}_{z \sim q} \left[\log \frac{q(z)}{f(z)} \right]. \tag{19}$$

If we are able to compute $\nabla_z f(z)$, then we can reparameterize (Kingma & Welling, 2013) in terms of the base density to obtain

$$\nabla_\theta D_{\text{KL}} \left(q_\theta(z) \| \tilde{f}(z) \right) = - \mathbb{E}_{u \sim \pi_0} \left[\nabla_\theta \log \left| J_{T_\theta(u)} \right| \right] \tag{20}$$

$$- \mathbb{E}_{u \sim \pi_0} \left[\nabla_\theta \log f(T_\theta(u)) \right] \tag{21}$$

where π_0 is the uniform distribution on the unit cube, and $J_{T_\theta(u)}$ is the Jacobian of T at u. The requirement of being able to compute $\nabla_z f(z)$ does not seem particularly onerous given that we assume we have a method to compute $f(z)$ and the mature tools available for automatic differentiation (Maclaurin et al., 2015).

However, if we cannot evaluate $\nabla_z f(z)$, then we must use the higher-variance score function estimator (Williams, 1992), with a gradient estimator $-\mathbb{E}_{z\sim q_\theta}[\log(f(z)/q_\theta(z))\nabla_\theta \log q_\theta(z)]$, using the reverse Jacobian to compute $q_\theta(z)$ and $\nabla_\theta \log q_\theta(z)$. Since our approach would then require evaluation of both T and T^{-1}, this would be slow for flow architectures such as autoregressive models.

4.3 JOINT TRAINING

Until now we have only discussed finding good partitions P_N and variational functions q_i separately. However, we might find that optimization difficulties or limited model expressiveness result in a partition which is not well-suited for our task. For example, if the flow collapses to a single mode of a multimodal distribution, most of the cells would be in that mode, resulting in a high variance estimator. Exactly matching the density of f is certainly a sufficient condition for providing a partitioning with minimal variance, but it is not necessary. Indeed when evaluating $\mathcal{L}_N$ using algorithm 1, any partitioning which splits up $\mathcal{Z}$ into regions of equal f density will work equally well given flexible q_i. We want to obtain our partitioning flow by solving the easiest possible problem, which raises the question of whether it is possible to directly optimize the flow parameters for the task of providing good partition cells, not simply density matching.

The most straightforward approach would be to directly optimize $\mathcal{L}_N$ from estimates using equation 10. However, the exponentials involved give the gradient a very high variance, with almost all of the gradient contribution to the base flow's parameters coming from the cell which happens to have a higher ELBO than the rest. This continues until the cell with the largest ELBO has become very large. Taking a more heuristic approach, we could also sample cells and optimize the mean ELBO. However, since we do not sample all the cells and the non-active cells do not contribute to the objective being optimized, there is a danger of the active cells expanding to take all of the density of f, essentially overfitting the active cells onto the density. We found that we could regularize this optimization procedure by including a term in the loss equal to the ELBO using a partition with only a single cell, with a variational density provided by the uniform distribution over the flow base space. In other words, the regularization term is the ELBO using the partitioning flow as a variational approximation on the whole space. Since the ELBO over the whole space will suffer if individual cells become too large, including the ELBO over the whole space penalises the behaviour of cells growing too large. Therefore the surrogate loss we use in training with n cells is

$$\frac{1-\lambda}{n}\sum_i \mathrm{ELBO}(\mathcal{B}_i) + \lambda\,\mathrm{ELBO}(\mathcal{Z}), \tag{22}$$

where $\mathrm{ELBO}(\mathcal{Z})$ is the ELBO taken over the whole space. Figure 3 shows a comparison of the two methods on a two-dimensional mixture-of-sixteen-Gaussians objective. We see that the plain variational approach has large areas of the support of the objective where there is little support from the variational function, and has some partition cells with quite pathologically stretched out shapes, while the method trained with the joint approach covers all of the modes.

4.4 IMPLEMENTATION DETAILS

We implemented our method with the RealNVP normalizing flow archtitecture (Dinh et al., 2017), using four layers for the partitioning normalizing flow and another four layers to model each of the instantiated variational distributions q_i. Each scale-and-shift function was modelled with a 256-hidden-unit multilayer perceptron with ReLU nonlinearities. Since a scale-and-shift from a unit cube does not result in support over $\mathbb{R}^d$, we added an elementwise inverse sigmoid operation before the first scale-and-shift, updating the log Jacobian accordingly. To model the densities in individual cells, we used a flow with an elementwise sigmoid after the final shift-and-scale layer, mapping into the unit box, and then scaled and shifted the output into the required cell, again incorporating the shift into the total log Jacobian. We incorporated a small $\epsilon = 10^{-5}$ into the sigmoid (i.e. mapping to $[\epsilon, 1-\epsilon]^d$ instead of $(0,1)^d$) to avoid numerical instability from samples that were very near the box boundary and so mapped to areas with extremely low objective density.

5 EXPERIMENTS

5.1 GAUSSIAN GRIDS

A setting where we expect our approach to help is the situation when the target density f has multiple modes. It has been noted that normalizing flows struggle with multi-modal densities (Dinh et al., 2019), and we would hope that our model would be able to focus on particular modes in each cell.

We generate a mixture-of-Gaussians density, with modes evenly spaced between -1 and 1, in four to twelve dimen-

Algorithm 2: Stratified Normalizing Flow

```
Input : Gradient-based update step, number of cells in
        partition $N$, number of cells to sample $n$, mixing
        parameter $\lambda$
Initialize parameters $\theta$ for a flow model with
transformation $T_\theta : \mathbb{R}^d \to \mathbb{R}^d$
// Train the normalizing flow to give the best partition
for $k$ Training Steps do
    Sample $n$ cells, $C_i$, randomly from the uniform
    partition of $(0,1)^d$
    Initialize $n$ flow models giving distributions $q_{\phi_i}$ on
    $C_i$, let $q_{\theta,\phi i}$ be the distribution on $T_\theta(C_i)$ induced by
    transforming $q_{\phi_i}$ through $T_\theta$
    for $m$ Inner Training Steps do
        Compute ELBO$_i$ from $q_{\theta,\phi_i}$ for each $i$
        Compute ELBO$_0$ induced by the uniform
        distribution on $(0,1)^d$ transformed through $T_\theta$
        Let $R = \lambda \cdot \text{ELBO}_0 + \frac{(1-\lambda)}{n}\sum_i \text{ELBO}_i$
        step $\phi_i$ and $\theta$ in the direction of $\nabla_{\phi_i} R$, $\nabla_\theta R$
    end
end
Use Algorithm 1 with pretrained partitioning flow
parameters $\theta$ to obtain $\hat{\mathcal{L}}_N$
```

sions. We use target distributions with two modes per side, and four modes per side. These are highly multi-modal distribution, with 65,536 modes in the eight-dimensional case with four modes per side. Each Gaussian is isotropic with variance 0.01 for the four mode per side case and 0.09 for the two mode per side case, ensuring that the modes are well-separated. We train our method with a base flow with 4 layers and instantiated flows of 4 layers, and compare to the ELBOs formed by training flow models on the reverse KL. We show results for a variational model using four and eight layers for comparison. The variational normalizing flows were trained using the Adam optimizer (Kingma & Ba, 2015) with step-size 10^{-5} for 10^5 iterations after an initial hyperparameter sweep. For our partitioning scheme, we used a grid side length of 0.5 in the base space, resulting in 2^d cells in dimension d. All the methods were implemented using the JAX differentiable programming library (Bradbury et al., 2018).[1]

The results are shown in figure 4. The true ELBO in all cases is equal to zero since the target is a properly normalized probability density. We see that all the models drop in performance as the dimension increases, probably due to the increasingly multimodal density. The variational models all perform better with two modes per side, likely

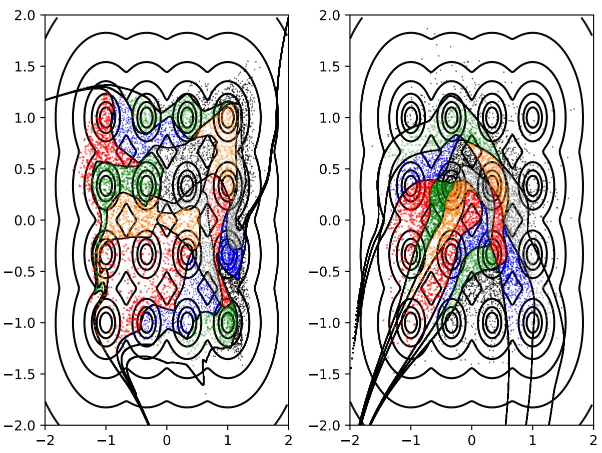

Figure 3: On this mixture-of-Gaussians problem, the right shows a flow model fitted variationally. On the left is the learned flow model for our approach. Our approach seems to lead to more mode-covering behaviour.

due to the increased multi-modality of the four-per-side distribution. However, our approach maintains more accuracy as the dimension increases, and is consistantly more accurate than the straightforward variational approach. As expected, we also observe the increased variance of our method as the number of cells increases, becoming noticably larger in ten, eleven and twelve dimensions.

5.2 BAYESIAN LINEAR REGRESSION/HIERARCHICAL MODEL

As a more realistic example, we consider marginalization in the setting of Bayesian linear regression. We generate a set of points according to the following probabilistic model: first an indicator $h \in \{1,2,3,4\}$ is drawn uniformly, indicating an assignment to one of four lines. Then y is sampled conditional on x and the parameters with probability

$$p(y|\boldsymbol{a},\boldsymbol{b},x,h) = \sum_i \mathcal{N}\left(a_i x + b_i, \sigma^2\right)\mathbb{I}(h=i), \quad (23)$$

with $\mathbb{I}(h=i)$ the 1-0 indicator function. We have a fixed known $\sigma = 0.1$. We generate a set of 80 data points $(\boldsymbol{x},\boldsymbol{y})$, with (a,b) pairs $(0,2),(1,0),(-1,-1),(0.5,2)$. The data is plotted in figure 5. We use a Gaussian prior on a_i and b_i with standard deviation equal to 3.

To perform posterior inference given the data we need to marginalise out variables by computing integrals. For instance,

$$P(a_1,b_1|\boldsymbol{y},\boldsymbol{x}) = \frac{P(\boldsymbol{y}|a_1,b_1,\boldsymbol{x})P(a_1,b_1)}{\int P(\boldsymbol{y}|\boldsymbol{a},\boldsymbol{b},\boldsymbol{x},h)P(\boldsymbol{a},\boldsymbol{b},h)d\boldsymbol{a}d\boldsymbol{b}dh}, \quad (24)$$

[1]Code is available at `https://github.com/ermongroup/stratified-flows`

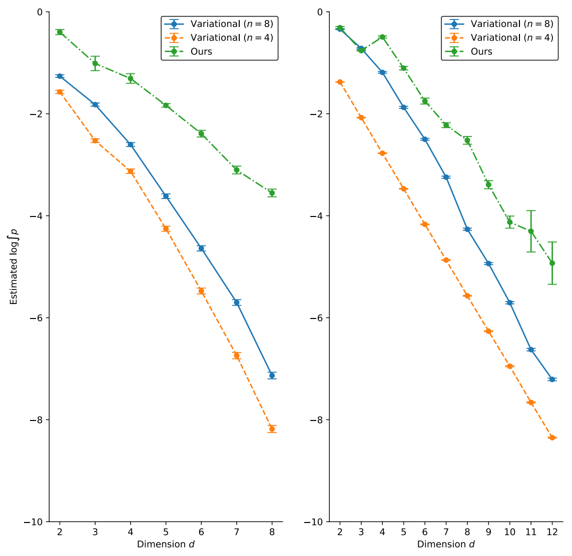

Figure 4: Showing the ELBOs calculated by our compared to the ELBO obtained by fitting the same architecture variationally, on a mixture-of-Gaussians objective. Left: 4^d modes. Right: 2^d modes. In all cases the ground truth value is 0.

	$\log P_1$	$\log P_2$
Uniform Sampling	-159 ± 4	-173 ± 6
Importance Sampling	-162.5 ± 0.1	-163.04 ± 0.04
Stratified Flow	-156.5 ± 0.4	-158.7 ± 0.42
Variational ($n = 4$)	-232.6 ± 0.9	-234.7 ± 0.9
Variational ($n = 8$)	-232.4 ± 0.7	-234.6 ± 0.8

Table 1: Results of calculating two probabilistic posteriors using various models, where $P_1 = P(a_1 = 0, b_1 = 2|\boldsymbol{y}, \boldsymbol{x})$, $P_2 = P(b_1 = 2|\boldsymbol{y}, \boldsymbol{x})$

with $P(\boldsymbol{y}|a_1, b_1, \boldsymbol{x})$ itself requiring marginalisation over all other slopes and intercepts and latent groupings h. To simplify the integration we absorb the marginalisation over h implicitly into the objective, with likelihood function $\frac{1}{4}\sum_i \mathcal{N}(a_i x + b_i, \sigma^2)$. To compare our method with other approaches, we imagine that we want to compute $P(a_1|b_1 = 2, \boldsymbol{y}, \boldsymbol{x})$. Doing this requires computing $P(a_1, b_1 = 2|\boldsymbol{y}, \boldsymbol{x})$ and $P(b_1 = 2|\boldsymbol{y}, \boldsymbol{x})$. The first term is a six-dimensional integral over $(a_2, a_3, a_4, b_2, b_3, b_4)$. The second is a 7-dimensional integral over $(a_1, a_2, a_3, a_4, b_2, b_3, b_4)$. Since we have a reasonable number of data points, we expect that the posterior has high density around the ground-truth values for the parameters. However, the posterior density is not unimodal, since with our data-generating process, the four different (a, b) pairs are not identifiable: we can swap the values of e.g. (a_1, b_1) with (a_2, b_2) and obtain the same likelihood for observed data. Since we fix $b_2 = 2$, this means there are six equivalent maximum-likelihood solutions under an infinite amount of data. There is also an additional set of high-likelihood parameters that can be seen from figure 5. Since the two lower lines are closer together, they can be explained by a single line which is close to horizontal and intercepts at around -2. The remaining two sets of parameters then become two degenerate lines which lie on top of each other with intercept zero. This solution also has equivalent modes–three in this case. As we show later, this mode is the mode that the variational methods collapse onto.

5.2.1 RESULTS

We compare the performance of the variational method with four layers, the variational method with eight layers and our approach. Since we cannot analytically evaluate the posterior, we do not know the exact ground truth values that we are hoping to approximate. To obtain an accurate benchmark, we construct an importance sampling distribution with a mixture of isotropic Gaussians, with one centered on each of the six permutations of ground truth parameters, taking 10 million samples from the proposal distribution for estimation. We also do a brute-force method of sampling with the uniform distribution using 200 million points. We expect that the uniform method has quite poor coverage of the space, since covering the $[-2, 2]^7$ box with spacing 0.1 would require over a hundred billion points.

We compute some of these integrals and compare the results in table 1. The results our method obtains are close to the importance sampled answer, and in fact on both computations, the stratified flow obtains a lower log-probability than the sampling methods. We believe this arises due to the importance sampling method having poor support on the extra solution with two degenerate lines. Since the IS distribution q is a mixture of Gaussians, the chance of getting a point far outside the ground-truth modes is very small (although compensated in expectation by a very large weight $1/q$). So we will tend to observe underestimation of the probability mass (Smith, 2001). With the reduced dimensionality of the 6-dimensional problem, we see that the uniform sampling method is closer to our method. This makes sense, as the same number of points achieve a much denser average packing in lower dimensions, and so give a more accurate result. Finally, in the appendix we analyse the failure of the variational methods, showing slices of the density assigned by the variational approaches. From these plots it is clear that the variational approach is identifying only a single mode of the distribution, which is the one where a single line has zero slope at intercept -1.5, and two lines have slope 1 at an intercept of zero.

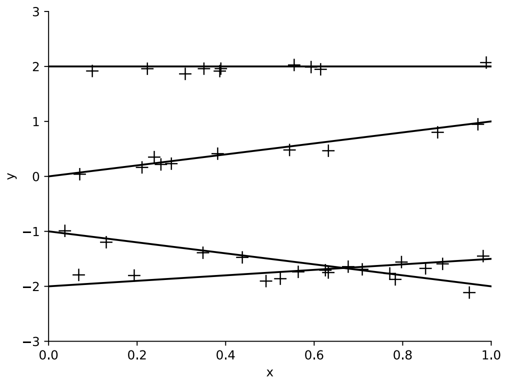

Figure 5: The data distribution used in the Bayesian marginalization task, and the lines corresponding to the ground-truth parameters.

6 RELATED WORK

6.1 FLEXIBLE IMPORTANCE SAMPLING

There is a large body of work on finding good importance sampling distributions (Cappé et al., 2004; Liang, 2002; Martino et al., 2015). In general, importance sampling proposal distributions require fast sampling of a sample $z \sim q$ and fast exact evaluation of $q(z)$, requirements which are naturally satisfied by normalizing flows. Previous work has investigated the application of flows to importance sampling, with an application to ray tracing in animation (Müller et al., 2019).

6.2 FLOWS AND MULTIMODALITY

The Real And Discrete (RAD) architecture (Dinh et al., 2019) combines a continuous flow with a categorical distribution. This allows separate modes to be assigned to different values of the categorical distribution, obtaining better log-likelihoods for multi-modal density estimation tasks such as the two-moons dataset. Similarly, the Localised Generative Flow family of models introduced in Cornish et al. (2019) are comprised of a collection of flows, where each individual flow only has to learn a local region of the objective function. They demonstrate that Masked Autoregressive Flows (Papamakarios et al., 2017) fail to learn a probability distribution comprised of two separated uniform distributions: the fact that the transform used is bijective means that some density must exist in a path between modes, leading to pathological distributions and unstable learning.

6.3 ADAPTIVE INFERENCE METHODS

Deliberately over or under-sampling regions of parameter space in order to reduce variance is a mature statistical technique known as stratified sampling (Neyman, 1934). A key aim of our method is to use the samples collected so far to learn which regions to over- or under-sample. This idea has a long history, with initial work such as the VEGAS algorithm (Lepage, 1978) significantly speeding up high-dimensional integration by forming a proposal distribution using histograms over the input space, fitted with the observed density. Adaptive inference methods can fail to converge on the correct value (Cappé et al., 2008) since they generally use samples from the previous proposal distributions to fit the next proposal distribution. Recent work introduces inference trees (Rainforth et al., 2018), a method for recursive partitioning the parameter space to give guarantees on the result obtained.

7 DISCUSSION AND CONCLUSION

We have introduced the stratified normalizing flow approach to approximate inference, allowing us to trade off between the high-bias, low-variance variational methods and the low-bias, high-variance Monte-Carlo methods with an approach that fits variational approximations inside a set of learned partition cells. In the limit of a single partition we recover the standard variational approach, while with an infinite number of cells our approach is equivalent to importance sampling. Through experiments on a highly multimodal toy problem and a more realistic Bayesian inference problem, we observe that this gives more accurate inference. We avoid the mode-collapsing tendency of variational methods (Bishop, 2006) while retaining their relatively quick optimization and good performance, but can reduce their bias by moving towards the importance sampling case by increasing the number of cells used. There are many possible avenues for further work. For instance, how exactly does the bias-variance tradeoff evolve as we increase the number of cells–is there an optimal size for the partition cells? It is plausible that the number of partition cells needed could be learned through optimization–perhaps by encoding the number of cells as a random variable and using a continuous relaxation of this (Maddison et al., 2017) to learn the best distribution.

Acknowledgements

This research was supported by NSF (#1651565, #1522054, #1733686), ONR (N00014-19-1-2145), AFOSR (FA9550-19-1-0024), FLI, and the Stanford Catalyst for Collaborative Solutions program.

References

Bishop, C. M. *Pattern recognition and machine learning*. Springer, 2006.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., and Wanderman-Milne, S. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/google/jax`.

Cappé, O., Guillin, A., Marin, J.-M., and Robert, C. P. Population monte carlo. *Journal of Computational and Graphical Statistics*, 13(4):907–929, 2004.

Cappé, O., Douc, R., Guillin, A., Marin, J.-M., and Robert, C. P. Adaptive importance sampling in general mixture classes. *Statistics and Computing*, 18(4): 447–459, 2008.

Cornish, R., Caterini, A. L., Deligiannidis, G., and Doucet, A. Localised generative flows. *arXiv preprint arXiv:1909.13833*, 2019.

Dick, J., Kuo, F. Y., and Sloan, I. H. High-dimensional integration: the quasi-monte carlo way. *Acta Numerica*, 22:133–288, 2013.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real nvp. *International Conference on Learning Representations*, 2017.

Dinh, L., Sohl-Dickstein, J., Pascanu, R., and Larochelle, H. A rad approach to deep mixture models. *ICLR Workshop on Deep Generative Models for Highly Structured Data*, 2019.

Grathwohl, W., Chen, R. T., Bettencourt, J., Sutskever, I., and Duvenaud, D. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *International Conference on Learning Representations*, 2019.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *International Conference on Learning Representations*, 2013.

Lepage, G. P. A new algorithm for adaptive multidimensional integration. *Journal of Computational Physics*, 27(2):192–203, 1978.

Liang, F. Dynamically weighted importance sampling in monte carlo computation. *Journal of the American Statistical Association*, 97(459):807–821, 2002.

Maclaurin, D., Duvenaud, D., and Adams, R. P. Autograd: Effortless gradients in numpy. *ICML Workshop on AutoML*, 2015.

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

Martino, L., Elvira, V., Luengo, D., and Corander, J. An adaptive population importance sampler: Learning from uncertainty. *IEEE Transactions on Signal Processing*, 63(16):4422–4437, 2015.

Müller, T., McWilliams, B., Rousselle, F., Gross, M., and Novák, J. Neural importance sampling. *ACM Transactions on Graphics*, 2019.

Neyman, J. On the two different aspects of the representative method: The method of stratified sampling and the method of purposive selection. *Journal of the Royal Statistical Society*, 97(4):558–625, 1934.

Owen, A. and Zhou, Y. Safe and effective importance sampling. *Journal of the American Statistical Association*, 95(449):135–143, 2000.

Papageorgiou, A. and Wasilkowski, G. W. On the average complexity of multivariate problems. *Journal of Complexity*, 6(1):1–23, 1990.

Papamakarios, G., Pavlakou, T., and Murray, I. Masked autoregressive flow for density estimation. In *Advances in Neural Information Processing Systems*, pp. 2338–2347, 2017.

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. Normalizing flows for probabilistic modeling and inference. *arXiv preprint arXiv:1912.02762*, 2019.

Rainforth, T., Zhou, Y., Lu, X., Teh, Y. W., Wood, F., Yang, H., and van de Meent, J.-W. Inference trees: Adaptive inference with exploration. *arXiv preprint arXiv:1806.09550*, 2018.

Ranganath, R., Gerrish, S., and Blei, D. M. Black box variational inference. *AISTATS*, 2013.

Rezende, D. J. and Mohamed, S. Variational inference with normalizing flows. *International Conference on Machine Learning*, 2015.

Smith, P. J. Underestimation of rare event probabilities in importance sampling simulations. *Simulation*, 76(3): 140–150, 2001.

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

Appendix

A Omitted Proofs

Theorem 3. *There is a number of samples n such that with probability greater than $1-\delta$, the empirical estimate $\hat{\mathcal{L}}_N$ defined in equation (13) is a lower bound to $\int_{\mathcal{Z}} f(z')dz'$.*

Proof. We have the empirical estimate

$$\hat{\mathcal{L}}_N = \frac{N}{n}\sum_{i=1}^{n} \exp\left[\mathbb{E}_{z\sim q_i}\left[\log f(z) - \log q_i(z)\right]\right], \tag{25}$$

where q_i is a density restricted to the partition $\mathcal{B}_i$ and the partitions $\mathcal{B}_i$ are sampled uniformly. For notational simplicity, let $F = \int_{\mathcal{Z}} f(z')dz'$.

Assume momentarily that we can compute the ELBO in each partition exactly. We have $0 \leq \exp \mathrm{ELBO}(\mathcal{B}_i) \leq F$, since the lower bound to the density in the cell cannot be negative and is less than the total density. Absorbing the N factor, we then have an average of a number of random variables, each in $[0, NF]$. From Popoviciu's inequality we have that the variance of a random variable in $[a, b]$ is upper-bounded by $\frac{1}{4}(a-b)^2$, and so we have

$$\mathrm{Var}(\hat{\mathcal{L}}_N) \leq \frac{N^2}{4n}F^2. \tag{26}$$

By Chebyschev's inequality we have

$$P(|\hat{\mathcal{L}}_N - \mathcal{L}_N| \geq \epsilon) \leq \frac{\mathrm{Var}[\hat{\mathcal{L}}_N]}{\epsilon^2} \tag{27}$$

$$= \frac{(N^2F^2}{4n\epsilon^2} \tag{28}$$

Plugging in $\epsilon = F - \mathcal{L}_N$, we have

$$1-\delta = \frac{N^2F^2}{4n(F-\mathcal{L}_N)^2}, \tag{29}$$

and so for $\hat{\mathcal{L}}_N$ to be a lower bound with probability greater than $1-\delta$ we need

$$n \geq \frac{N^2F^2}{(1-\delta)(F-\mathcal{L}_N)^2} \tag{30}$$

$\square$

In the case with non-exact ELBOs, if we know that with high probability the ELBO estimates are good enough to be less than F (which for a reasonable number of cells is likely to be the case) then we can union bound over the individual ELBOs. We then obtain the result that $\hat{\mathcal{L}}_N$ is a lower bound with high probability, with a few extra factors due to the variance in the ELBOs.

B Density plots for Bayesian linear regression experiment

These plots show the density of the distribution induced by the $(n = 8)$ flow model trained variationally on the objective function $P(\boldsymbol{y}, b_1|\boldsymbol{x})$ as described in the main text. To generate these plots, we sample 10,000 points from the base distribution, push through the transforming function and plot the densities of the resulting points. We plot the projection of the points on two-dimensional planes corresponding to individual a_i, b_i pairs, as the density in those planes is interpretable. Also shown is the one-dimensional marginal corresponding to a_1. We also show a ground-truth plot of the high-density regions, obtained by sampling uniformly from the parameter space and discarding points with log-likelihood under a certain threshold (-370 for our case). In all cases we observe that the densities are in anticorrelated ellipses, since to a first order approximation, increasing the intercept can be compensated by reducing the slope.

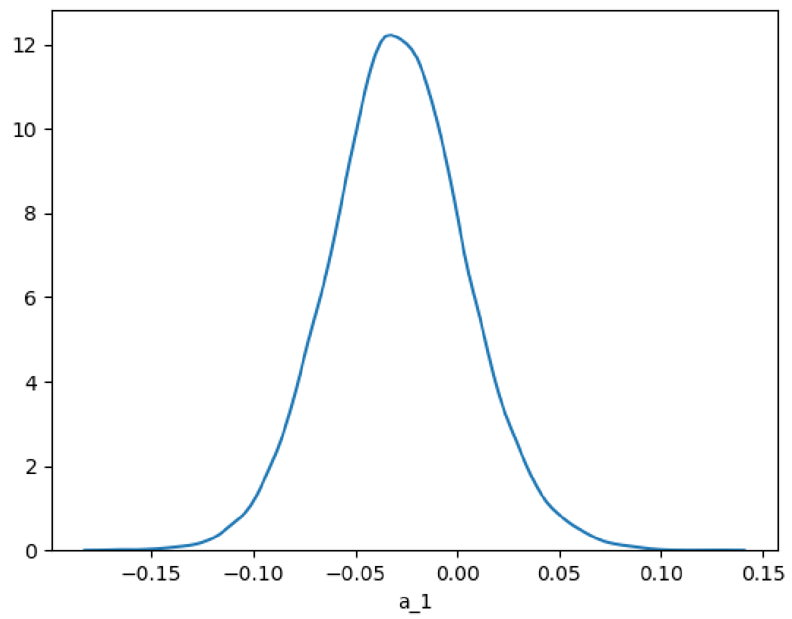

Figure 6: Showing the marginal distribution of a_1 given by the flow model. The density is clustered around the ground-truth.

C EXTENDED ALGORITHM FOR ALGORITHM 2

We give a slightly longer version of the algorithm 2 listing that we omit for space in the main text.

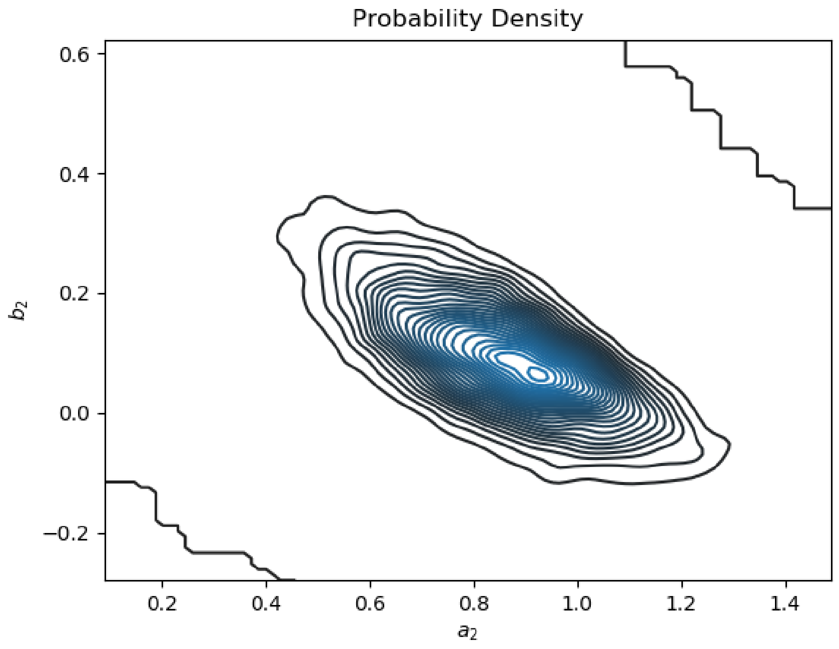

Figure 7: Showing the marginal distribution of a_2, b_2 given by the variationally trained flow model. It is clustered around an intercept of 0 and a slope of 0.9, corresponding to the real ground truth values of 0 and 1.

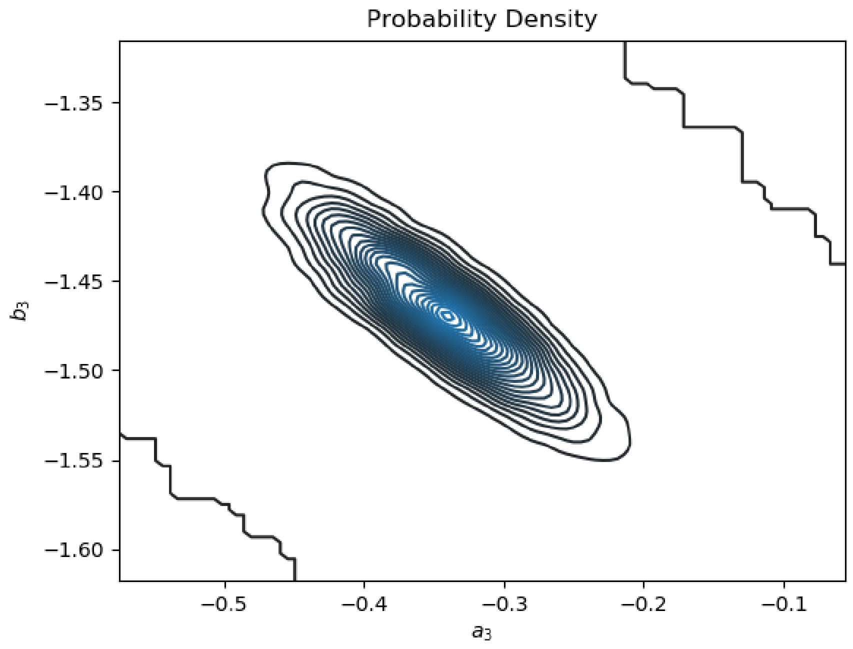

Figure 8: Showing the marginal distribution of a_3, b_3 given by the variationally trained flow model. It is clustered around an intercept of -1.5 and a slope of -0.3, which doesn't correspond to any of the ground truth values. However, looking at the plot in figure 4 in the main text, we see that this line can plausibly explain the two lines with intercept -1 and -2.

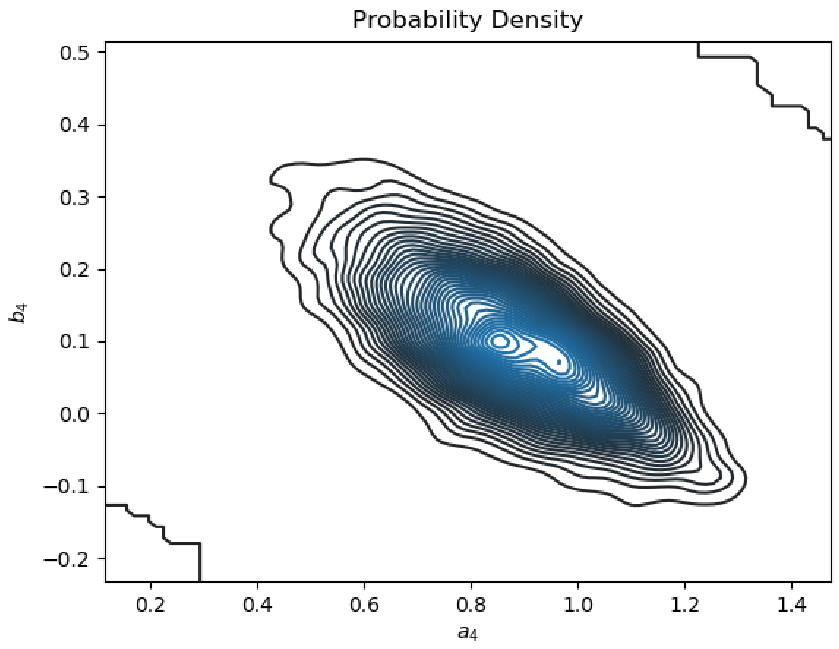

Figure 9: Showing the marginal distribution of a_4, b_4 given by the variationally trained flow model. This is clustered around a slope of 0.9 and an intercept of 0.1, which corresponds to the line with ground truth values 1.0 and 0, in essentially the same configuration as the a_2, b_2 plane density.

Algorithm 3: Stratified Normalizing Flow

```
Input :Parameters $\theta$ for a flow model with transformation $T_\theta : \mathbb{R}^d \to \mathbb{R}^d$, gradient-based optimization update step,
       number of cells in partition $N$, number of cells to sample $n$, training batch size $h$
for $k$ Training Steps do
    Sample $n$ cells, $C_i$, randomly from the uniform partition of $(0, 1)^d$
    Initialize $n$ flow models $T_{\phi_i} : \mathbb{R}^d \to C_i$ with parameters $\phi_i$, let $q_{\theta,\phi i}$ be the distribution on $T_\theta(C_i)$ from the flow
    model with combined transformation $T_\theta \circ T_{\phi_i}$
    for $m$ Inner Training Steps do
        Sample $u_{1:h} \sim \mathcal{U}((0, 1)^d)$,
        Let $E_0 = \frac{1}{h} \sum_j \log f(T_\theta(u_j)) - \log q_\theta(u_j)$
        for $i \in (1, \ldots, n)$ do
            Sample $u_{1:h} \sim \mathcal{U}(C_i)$, $j \in [k]$
            Let $E_i = \frac{1}{h} \sum_j \log f(T_\theta \circ T_{\phi_i}(u_j)) - \log q_{\theta,\phi_i}(u_j)$
        end
        Let $R = \lambda \cdot E_0 + \frac{(1-\lambda)}{n} \sum_i E_i$
        $\theta \leftarrow \mathtt{step}(\theta, \nabla_\theta R)$
        $\phi_i \leftarrow \mathtt{step}(\phi_i, \nabla_{\phi_i} R)$, for $i \in (1, \ldots, n)$
    end
end
Use Algorithm 1 with pretrained partitioning flow parameters $\theta$ to obtain $\hat{\mathcal{L}}_N$
```

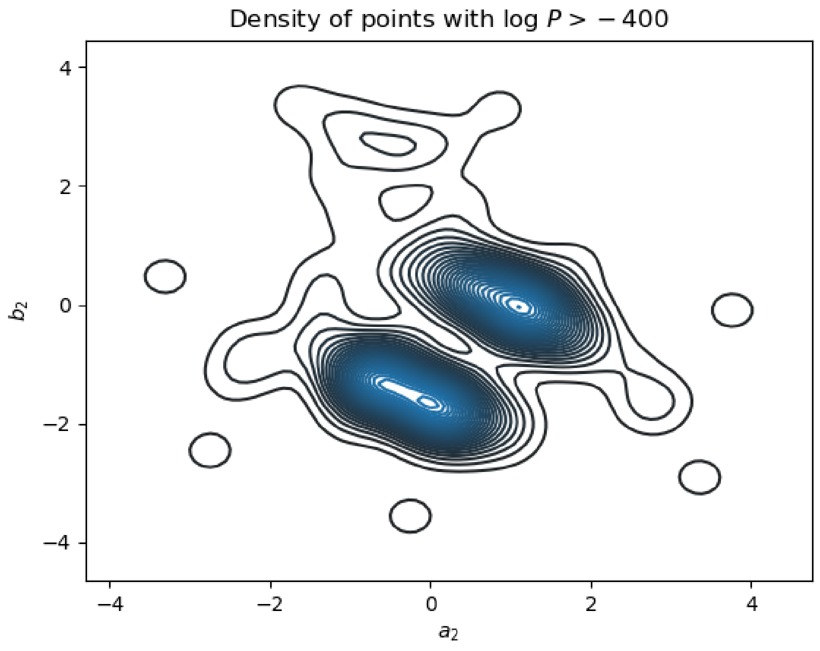

Figure 10: Showing the ground-truth high density regions of the objective function obtained by brute-force sampling in the a_2, b_2 plane. We observe two separate regions of high density. The region at $(1, 0)$ corresponds to the ground truth value $(1, 0)$, while the extended region from $(1, -2)$ to $(-1, -1)$ corresponds to the ground truth values $(1, -2)$ and $(-1, -1)$. Since the lines are close together, the two regions overlap significantly. The multi-modality is clearly observable from this plot, which the variational flow misses.