Bootstrap Your Flow

Laurence Illing Midgley laurencemidgley@gmail.com
Vincent Stimper vs488@cam.ac.uk
Gregor N. C. Simm gncs2@cam.ac.uk
José Miguel Hernández-Lobato jmh233@cam.ac.uk
Department of Engineering, University of Cambridge

Abstract

Normalizing flows are flexible, parameterized distributions that can be used to approximate expectations from intractable distributions via importance sampling. However, current flow-based approaches are limited on challenging targets where they either suffer from mode seeking behaviour or high variance in the training loss, or rely on samples from the target distribution, which may not be available. To address these challenges, we combine flows with annealed importance sampling (AIS), while using the α-divergence as our objective, in a novel training procedure, FAB (Flow AIS Bootstrap). Thereby, the flow and AIS improve each other in a bootstrapping manner. We demonstrate that FAB can be used to produce accurate approximations to complex target distributions, including Boltzmann distributions, in problems where previous flow-based methods fail. We provide code for our experiments at https://github.com/lollcat/FAB-2021.

1. Introduction

Estimating expectations with respect to target distributions that cannot be sampled from is a challenging task with many real world applications, such as estimating equilibrium properties of physical systems governed by the Boltzmann distribution (Lelièvre et al., 2010). Boltzmann generators (Noé et al., 2019), which use normalizing flows to approximate the Boltzmann distribution, are a recent approach with growing interest (Dibak et al., 2020; Köhler et al., 2021). For challenging problems, current approaches to training Boltzmann generators rely partly on samples from the target for training by the maximum likelihood Wu et al. (2020). Such samples are obtained through expensive Molecular Dynamics simulations (Leimkuhler and Matthews, 2015). Although flows can be trained without samples from the target, current methods for this suffer from either being mode seeking or high variance in the loss, which leads to inferior performance on challenging problems (Stimper et al., 2021).

To address these challenges, we propose using the α-divergence with $\alpha = 2$ as the training objective, which is mass covering, and employ annealed importance sampling (AIS) to bring the samples from the flow model closer to the target, reducing variance in the objective. In our experiments, we apply our method, **F**low **A**IS **B**ootstrap (**FAB**), to a 2D Gaussian mixture distribution as well as to the "Many Well" problem and show that it outperforms competing learning algorithms.

Although we focus on a toy problem and the Boltzmann distribution, our approach for training flows to approximate intractable distributions is not specific to these problems,

© L.I. Midgley, V. Stimper, G. N. C. Simm & J.M. Hernández-Lobato.

and can be applied to any target distribution for which an un-normalized probability density function is defined. One key application of interest that we leave for future work is approximate inference over Bayesian posterior distributions (for example the posterior of a Bayesian Neural Network). In the context of Bayesian inference, in terms of computational complexity, FAB lies between Markov chain Monte Carlo, the computationally expensive gold-standard for Bayesian inference, and Bayes-by-backprop (Blundell et al., 2015) which provides a computationally cheaper alternative, but suffers from worse performance.

2. Background

Normalizing flows Given a random variable $\mathbf{z}$ with distribution $q(\mathbf{z})$ a normalizing flow (Rezende and Mohamed, 2015; Tabak and Vanden-Eijnden, 2010) uses an invertible map $F : \mathbb{R}^d \to \mathbb{R}^d$ to transform $\mathbf{z}$ yielding the random variable $\mathbf{x} = F(\mathbf{z})$ with the distribution

$$q(\mathbf{x}) = q(\mathbf{z}) \left|\det(J_F(\mathbf{z}))\right|^{-1}. \tag{1}$$

If we parameterize F we can use this as a model to approximate a target distribution p. If the target density $p(\mathbf{x})$ is available, the flow is usually trained by minimizing the KL divergence, which is estimated via Monte Carlo with samples from the flow model. Alternatively, we could use the α-divergence (Zhu and Rohwer, 1995), which is defined by

$$D_\alpha(p\|q) = \frac{\int_x \alpha p(\mathbf{x}) + (1-\alpha)q(\mathbf{x}) - p(\mathbf{x})^\alpha q(\mathbf{x})^{1-\alpha} d\mathbf{x}}{\alpha(1-\alpha)}, \tag{2}$$

as an objective (Campbell et al., 2021; Müller et al., 2019). If $\alpha = 2$, minimizing it corresponds to minimizing the variance of the importance weights $w_{\mathrm{IS}}(\mathbf{x}) = \frac{p(\mathbf{x})}{q(\mathbf{x})}$. In contrast to KL divergence, which is mode seeking, $D_{\alpha=2}(p\|q)$ is mass covering which is more desirable when approximating multimodal targets. In this case, the α-divergence can be rewritten as

$$D_{\alpha=2}(p\|q) \propto \int \frac{p(\mathbf{x})^2}{q(\mathbf{x})} d\mathbf{x} = \mathrm{E}_{q(\mathbf{x})}\left[w_{\mathrm{IS}}(\mathbf{x})^2\right] = \mathrm{E}_{p(\mathbf{x})}\left[w_{\mathrm{IS}}(\mathbf{x})\right]. \tag{3}$$

This objective can be estimated either with samples from $p(\mathbf{x})$ or $q(\mathbf{x})$. Since the integral is dominated by regions with high $p(\mathbf{x})$ and low $q(\mathbf{x})$, the estimate will exhibit higher variance if we sample from q than if we sample from p.

Annealed importance sampling AIS begins by sampling from an initial proposal distribution $\mathbf{x}_0 \sim p_0 = q$, being the flow in our case, and then transitions via MCMC through a sequence of intermediate distributions, p_1 to p_{N-1}, to produce samples $\mathbf{x}_{N-1}$ closer to the target distribution $p_N = p$ (Neal, 2001). Each transition T_j is a Markov chain that leaves the intermediate distribution p_j invariant. AIS conventionally returns the importance weights for the samples, which are in the form

$$w_{\mathrm{AIS}}(\mathbf{x}^{(i)}) = \frac{\tilde{p}_1(\mathbf{x}_0)}{p_0(\mathbf{x}_0)} \frac{\tilde{p}_2(\mathbf{x}_1)}{\tilde{p}_1(\mathbf{x}_1)} \cdots \frac{\tilde{p}_{N-1}(\mathbf{x}_{N-2})}{\tilde{p}_{N-2}(\mathbf{x}_{N-2})} \frac{\tilde{p}_N(\mathbf{x}_{N-1})}{\tilde{p}_{N-1}(\mathbf{x}_{N-1})} \tag{4}$$

where $\mathbf{x}^{(i)} = \mathbf{x}_{N-1}$ and we indicate that the probability density functions may be unnormalized with $\tilde{p}$. They exhibit variance reduction compared to their counterparts $w_{\mathrm{IS}}(\mathbf{x})$. Hamiltonian Monte Carlo (HMC) provides a suitable choice of transition operator for challenging problems (Neal, 1995).

3. Normalizing Flow Annealed Importance Sampling Bootstrap

FAB, defined in Algorithm 1, uses $\log(D_{\alpha=2}(p\|q))$ as a training objective. Furthermore, we introduce AIS into the training loop, improving the gradient estimator for minimising $D_{\alpha=2}(p\|q)$ by writing the loss function to train the flow as an expectation over $p(\mathbf{x})$, and estimating it with the samples and importance weights generated by AIS with the flow as the initial distribution. If we plug in (4) in (3), compute the expectation over $p(\mathbf{x})$ through AIS and use Jensen's inequality, we obtain our loss

$$\mathcal{L}(\theta) = \log \sum_{l=1}^{L} \exp\left(\log \bar{w}_{\text{AIS}}^{(l)} + \left(\log p(\bar{\mathbf{x}}_{\text{AIS}}^{(l)}) - \log q(\bar{\mathbf{x}}_{\text{AIS}}^{(l)})\right)\right), \tag{5}$$

where θ are the flow's parameter and $\bar{w}_{AIS}$ and $\bar{\mathbf{x}}_{AIS}$ have blocked gradients[1]. This method obtains the benefit of bootstrapping, where AIS is used to improve the flow by improved estimation of the loss's gradient, which is used to update the flow. This in turn improves AIS by improving its initial distribution. The effective sample size of the trained flow model, which may be limited e.g. through expressiveness constraints of the specific flow, can also be improved by using AIS after training.

Algorithm 1: FAB for minimisation of $D_{\alpha=2}(p\|q)$

Set target p
Initialise proposal q parameterized by θ
for *iteration = 1, M* **do**
  Sample batch $\mathbf{x}_q^{(1:L)}, \log q(\mathbf{x}_q^{(1:L)})$ from q
  Generate batch $\mathbf{x}_{AIS}^{(1:L)}$, $\log w_{AIS}^{(1:L)}$ from AIS seeded with $\mathbf{x}_q^{(1:L)}, \log q(\mathbf{x}_q^{(1:L)})$
  Calculate FAB loss: $\mathcal{L}(\theta)$
  Perform gradient descent on $\mathcal{L}(\theta)$
end

4. Experiments

4.1. Mixture of Gaussians Problem

We begin with a simple two dimensional mixture of Gaussians target distribution. To estimate expectations $\mathbb{E}_{p(\mathbf{x})}[f(\mathbf{x})]$ with our proposal distribution, we set $f(\mathbf{x})$ to be the toy quadratic function $f(\mathbf{x}) = \mathbf{a}^T(\mathbf{x} - 2\mathbf{b}) + 2(\mathbf{x} - 2\mathbf{b})^T \mathbf{C}(\mathbf{x} - 2\mathbf{b})$, where the elements of $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{C}$ are sampled from a unit Gaussian and then fixed for the problem. This allows us to inspect the bias and variance of estimates of the expectation of this toy function, as a further measure of performance. We compare FAB[2] to flows trained through minimisation

1. We only want to calculate the gradient with respect to the $\log(D_{\alpha=2}(p\|q))$, and not with respect to the importance weights and samples generated by AIS, which are also a function of the flow's parameters.
2. For all experiments, we use FAB with only 2 intermediate AIS distributions. For the transition operator between AIS distributions we use HMC with 1 outer step and 5 inner steps.

of KLD[3], as well as Stochastic Normalizing Flows (SNFs) (Wu et al., 2020). Like FAB, SNFs also combine flows with stochastic sampling methods such as MCMC, but instead focus on improving the flow's expressive power. The SNF is trained with the KL divergence as well. For all models we choose real NVP (Dinh et al., 2017) as the flow architecture.

In Figure 1 we see that FAB allows us to train a flow that captures the shape of the target distribution well, while the flow and SNF trained with KLD both fail, capturing only a subset of the modes. Table 1[4] shows that with FAB the trained flow may be used for accurate computation of expectations with respect to the target, while the alternative approaches yield highly biased estimates.

Table 1: Performance of FAB vs a flow and SNF trained with KLD on the Mixture of Gaussians problem. For FAB all metrics are for the trained flow, while the metrics after AIS are provided in brackets where applicable.

Model	Mean $\log q(\mathbf{x})$, $\mathbf{x} \sim p(\mathbf{x})$	ESS (%)	Bias (%)	Std (%)
FAB	-5.2	70.1 (77.5)	1.2 (0.5)	5.8 (5.5)
Flow trained with KLD	-14.4	0.05	99.6	19.8
SNF trained with KLD	N/A	0.02	104.2	9.7

4.2. The Many Well Problem

For a more challenging problem, we test FAB against a flow trained by KLD on a 16 dimensional "Many Well" problem, which we create by repeating the Double Well Boltzmann distribution from (Noé et al., 2019) 8 times. We create a hand-crafted test-set for this problem where we place a point on each of the 256 modes of the target. In Table 2 we see that FAB allows us to train a model that has a far superior test set log-likelihood and ESS than training a flow with KLD. In Figure 2 where we visualise a subset of the marginal distributions for pairs of dimensions, we see that the flow trained with FAB has captured the shape of the target well, while the flow trained with KLD fits only a subset of the modes.

Table 2: Performance of FAB vs a flow trained by KLD on the Many Well Problem. Metrics are provided with respect to the trained flows, while for FAB, ESS after AIS is provided in brackets.

Model	Test set mean $\log q(\mathbf{x})$	ESS (%)
FAB	-14.5	79.6 (85.2)
Flow trained by KLD	-86.2	0.01

3. For brevity we refer to a flow/SNF trained through minimisation of KL divergence, estimated with samples from the flow/SNF, as simply being "trained with KLD".
4. Effective sample size (ESS) for both experiments is calculated with 10^6 samples. Bias and standard deviation are calculated using 100 runs of 1000 samples. We calculate the mean target log likelihood using 10000 samples.

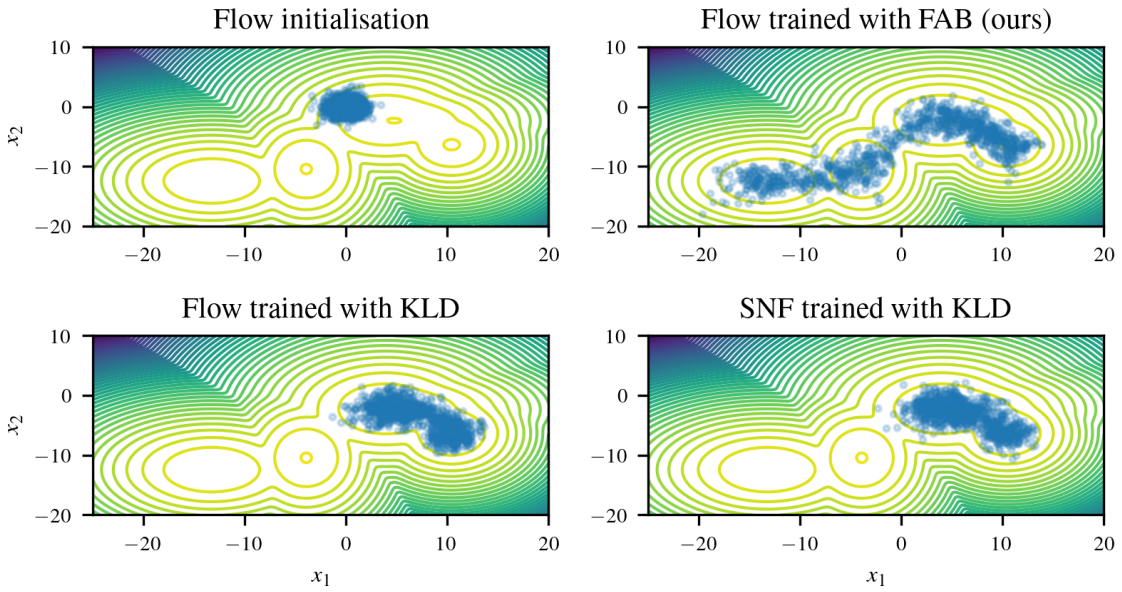

Figure 1: Mixture of Gaussians problem target probability contours, overlaid with samples from trained models. With FAB the flow captures all of the modes, while the flow and SNF trained with KLD fit a subset of the modes.

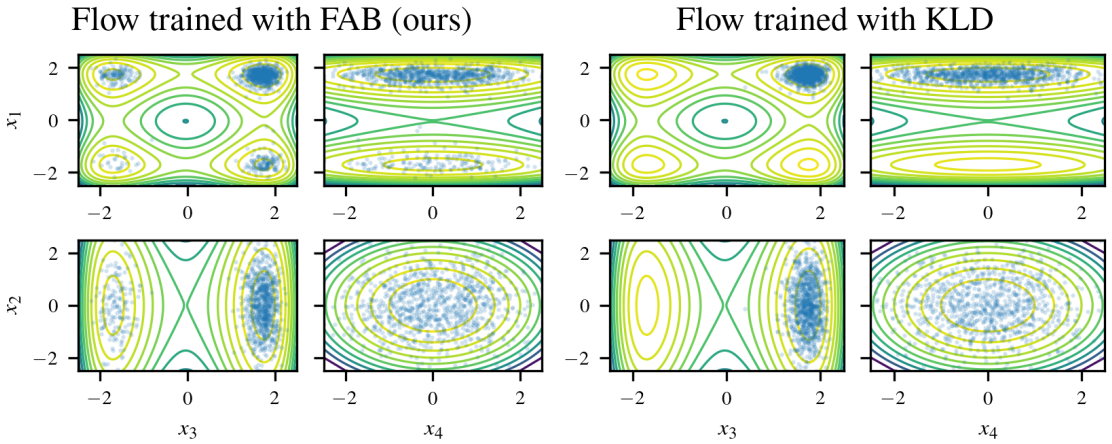

Figure 2: Target probability contours for pairs of marginal distributions for the first four dimensions of the 16 dimensional Many Well Problem, overlaid with samples from trained models. With FAB the flow captures all of the marginals' modes, while the flow trained with KLD, fits a subset of them.

5. Discussion

In SNFs, MCMC is combined with normalizing flows by introducing sampling layers between the standard flow layers to improve the expressiveness of the flow (Nielsen et al., 2020; Wu et al., 2020). They are usually trained with samples from the target and perform poorly when trained exclusively with samples from the flow, see section 4.1. Instead, FAB focuses on improving the training procedure. However, the two contributions are orthogonal, so SNFs could be trained with FAB as well. Similarly, FAB works with other flow architectures (Chen et al., 2019; Grcić et al., 2021) and base distributions (Papamakarios et al., 2017; Stimper et al., 2021) than those we used. In Ding and Freedman (2019) AIS is applied in the context of variational inference to improve estimation of the marginal likelihood gradient for varitational autoencoders' decoder. This approach may be extended by FAB to also improve the encoder training, through minimising the FAB loss with respect to the parameters of the encoder, with the latent posterior as the target distribution. This may be advantageous relative to (Ding and Freedman, 2019), as well as MCMC based approaches like (Hoffman, 2017) for training the decoder, because if the flow-based encoder can learn a good approximation of the latent posterior, then this alleviates the requirement to run MCMC for long in order to obtain good samples for the decoder. In neural transport MCMC (Hoffman et al., 2019), a flow is trained via KL divergence minimisation with a target distribution, and then is used to perform MCMC with the target in the latent space of the flow. This improves the geometry in which MCMC takes place, allowing for better mixing. Neural transport MCMC relies on the flow capturing the geometry of the target distribution and can worsen sampling in the tails of the target distribution if the flow fails to capture the target well. Thus by replacing the KL divergence training objective with FAB training, our approach may be useful for improving the robustness of neural transport MCMC.

We proposed FAB, a novel method of combing flows with AIS in a training procedure that allows them to improve each other in a bootstrapping manner. For future work we hope to (1) identify how important each component of FAB is, such as the number of intermediate AIS distributions used and the form of the loss (2) perform a rigorous benchmarking of FAB's performance and computational complexity relative to alternative approaches such as MCMC, (3) explore the aforementioned connections of FAB to other recent advances in literature, and (4) scale up FAB and apply it to to more challenging real world problems, for example Boltzmann distributions of complex molecules.

Acknowledgments

GNCS and JMHL acknowledge support from a Turing AI Fellowship under grant EP/V023756/1. This work has been performed using resources operated by the University of Cambridge Research Computing Service, which is funded by the EPSRC (capital grant EP/P020259/1) and DiRAC funding from the STFC (http://www.dirac.ac.uk/). LIM acknowledges support from the Cambridge Trust, the Skype Foundation, and the Oppenheimer Memorial Trust.

References

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *International Conference on Machine Learning*, pages 1613–1622. PMLR, 2015.

Andrew Campbell, Wenlong Chen, Vincent Stimper, Jose Miguel Hernandez-Lobato, and Yichuan Zhang. A gradient based strategy for hamiltonian monte carlo hyperparameter optimization. In *International Conference on Machine Learning*, pages 1238–1248. PMLR, 2021.

Ricky T. Q. Chen, Jens Behrmann, David Duvenaud, and Jörn-Henrik Jacobsen. Residual Flows for Invertible Generative Modeling. In *Advances in Neural Information Processing Systems*, volume 32, December 2019.

Manuel Dibak, Leon Klein, and Frank Noé. Temperature-steerable flows. *Third Workshop on Machine Learning and the Physical Sciences (NeurIPS 2020),*, December 2020.

Xinqiang Ding and David J Freedman. Learning deep generative models with annealed importance sampling. *arXiv preprint arXiv:1906.04904*, 2019.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv eprint arXiv:1605.08803*, 2017.

Matej Grcić, Ivan Grubišić, and Siniša Šegvić. Densely connected normalizing flows. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Matthew Hoffman, Pavel Sountsov, Joshua V Dillon, Ian Langmore, Dustin Tran, and Srinivas Vasudevan. Neutra-lizing bad geometry in hamiltonian monte carlo using neural transport. *arXiv preprint arXiv:1903.03704*, 2019.

Matthew D Hoffman. Learning deep latent gaussian models with markov chain monte carlo. In *International conference on machine learning*, pages 1510–1519. PMLR, 2017.

Jonas Köhler, Andreas Krämer, and Frank Noé. Smooth Normalizing Flows. In *Advances in Neural Information Processing Systems 34*, 2021.

Ben Leimkuhler and Carles Matthews. *Molecular Dynamics With Deterministic and Stochastic Numerical Methods*. Number 39 in Interdisciplinary Applied Methematics. Springer, 2015.

Tony Lelièvre, Mathias Rousset, and Gabriel Stoltz. *Free Energy Computations: A Mathematical Perspective*. IMPERIAL COLLEGE PRESS, June 2010. ISBN 978-1-84816-247-1 978-1-84816-248-8. doi: 10.1142/p579.

Thomas Müller, Brian Mcwilliams, Fabrice Rousselle, Markus Gross, and Jan Novák. Neural Importance Sampling. *ACM Transactions on Graphics*, 38(5):145:1–145:19, October 2019. ISSN 0730-0301. doi: 10.1145/3341156.

Radford M Neal. Annealed importance sampling. *Statistics and computing*, 11(2):125–139, 2001.

RM Neal. Bayesian learning for neural networks [phd thesis]. *Toronto, Ontario, Canada: Department of Computer Science, University of Toronto*, 1995.

Didrik Nielsen, Priyank Jaini, Emiel Hoogeboom, Ole Winther, and Max Welling. SurVAE Flows: Surjections to Bridge the Gap between VAEs and Flows. *Advances in Neural Information Processing Systems 33*, 2020.

Frank Noé, Simon Olsson, Jonas Köhler, and Hao Wu. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365(6457), 2019.

George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 2335–2344, Red Hook, NY, USA, December 2017. Curran Associates Inc.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

Vincent Stimper, Bernhard Schölkopf, and José Miguel Hernández-Lobato. Resampling Base Distributions of Normalizing Flows. *arXiv preprint arXiv:2110.15828*, 2021.

Esteban G. Tabak and Eric Vanden-Eijnden. Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, 8(1):217 – 233, 2010. doi: cms/1266935020. URL https://doi.org/.

Hao Wu, Jonas Köhler, and Frank Noe. Stochastic normalizing flows. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 5933–5944. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/41d80bfc327ef980528426fc810a6d7a-Paper.pdf.

Huaiyu Zhu and Richard Rohwer. Information geometric measurements of generalisation. Technical report, Aston University, 1995.

Appendix A. FAB derivation

FAB aims to minimise

$$D_{\alpha=2}(p\|q) \propto \int \frac{p(\mathbf{x})^2}{q(\mathbf{x})} d\mathbf{x} = \mathrm{E}_{p(\mathbf{x})}\left[\frac{p(\mathbf{x})}{q(\mathbf{x})}\right]. \tag{6}$$

We obtain a gradient estimator by differentiating Equation 6 with respect to the parameters θ of the flow model

$$\nabla_\theta \left[\mathrm{E}_{p(\mathbf{x})}\left[\frac{p(\mathbf{x})}{q_\theta(\mathbf{x})}\right]\right] = \mathrm{E}_{p(\mathbf{x})}\left[\nabla_\theta \frac{p(\mathbf{x})}{q_\theta(\mathbf{x})}\right], \tag{7}$$

where we have used the fact that since θ is independent to samples from $p(\mathbf{x})$, we can move ∇_θ inside the expectation. If we set $f(\mathbf{x}) = \nabla_\theta \frac{p(\mathbf{x})}{q_\theta(\mathbf{x})}$, we see that Equation 7 is in the form $\mathbb{E}_{p(\mathbf{x})}[f(\mathbf{x})]$, so we can estimate it with AIS.

With this goal in mind, during the training loop, we generate a batch of importance weights $w_{\text{AIS}}^{(1:L)}$, and samples $\mathbf{x}_{\text{AIS}}^{(1:L)}$ using AIS, with $p(\mathbf{x})$ as the target distribution and $q_\theta(\mathbf{x})$ as the proposal distribution. We can then obtain an importance weighted estimate of the above gradient operator

$$\mathrm{E}_{p(\mathbf{x})}\left[\nabla_\theta \frac{p(\mathbf{x})}{q_\theta(\mathbf{x})}\right] \underset{\sim}{\propto} \sum_{l=1}^{L} \bar{w}_{\text{AIS}}^{(l)} \left[\nabla_\theta \frac{p(\bar{\mathbf{x}}_{\text{AIS}}^{(l)})}{q_\theta(\bar{\mathbf{x}}_{\text{AIS}}^{(l)})}\right]. \tag{8}$$

We note that in Equation 7, $q_\theta(\mathbf{x})$ is only differentiated through θ's contribution to the probability density function, and not[5] via $\nabla_\theta \mathbf{x}$. Therefore, in Equation 8 we take care to block the gradient of $\mathbf{x}_{\text{AIS}}^{(1:L)}$ with respect to θ. We denote the blocked gradients with $\bar{\mathbf{x}}_{AIS}^{(l)}$. Thus, we can train the proposal by minimising the surrogate "loss function"

$$O(\theta) = \sum_{l=1}^{L} \bar{w}_{\text{AIS}}^{(l)} \left[\frac{p(\bar{\mathbf{x}}_{\text{AIS}}^{(l)})}{q_\theta(\bar{\mathbf{x}}_{\text{AIS}}^{(l)})}\right], \tag{9}$$

taking care[6] to block the gradient of $\bar{w}_{\text{AIS}}^{(1:L)}$ and $\bar{\mathbf{x}}_{\text{AIS}}^{(1:L)}$ with respect to θ.

To obtain a good loss function for training it is beneficial to instead seek to write the surrogate loss (Equation 9) in terms of log probabilities and log importance weights, because inside the expectation the importance weights and fractions of probabilities will have high variance. To do this we can re-write the surrogate loss as

$$\begin{aligned} O(\theta) &= \exp\log \sum_{l=1}^{L} \bar{w}_{\text{AIS}}^{(l)} \left[\frac{p(\bar{\mathbf{x}}_{\text{AIS}}^{(l)})}{q_\theta(\bar{\mathbf{x}}_{\text{AIS}}^{(l)})}\right] \\ &= \exp\log \sum_{l=1}^{L} \exp\Bigg(\log \bar{w}_{\text{AIS}}^{(l)} + \\ &\qquad \Big(\log p(\bar{\mathbf{x}}_{\text{AIS}}^{(l)}) - \log q_\theta(\bar{\mathbf{x}}_{\text{AIS}}^{(l)})\Big)\Bigg). \end{aligned} \tag{10}$$

Finally, we minimise the below loss, which by Jensen's inequality is an upper bound of $\log O(\theta)$,

$$\begin{aligned} \mathcal{L}(\theta) &= \log \sum_{l=1}^{L} \exp\Bigg(\log \bar{w}_{\text{AIS}}^{(l)} + \\ &\qquad \Big(\log p(\bar{\mathbf{x}}_{\text{AIS}}^{(l)}) - \log q_\theta(\bar{\mathbf{x}}_{\text{AIS}}^{(l)})\Big)\Bigg). \end{aligned} \tag{11}$$

We can now work with log probabilities and log importance weights, and use the "logsumexp" trick to obtain a numerically stable estimate. Equation 11 is the exact surrogate loss implemented in practice for training.

5. In Equation 7, ∇_θ is inside the expectation, with $\mathbf{x} \sim p(\mathbf{x})$ independent to θ.
6. In Equation 8, $w_{\text{AIS}}^{(1:L)}$ is not differentiated with respect to θ, so we must block the gradient of $w_{\text{AIS}}^{(1:L)}$ with respect to θ, as otherwise automatic differentiation will result in an incorrect estimate of the gradient. This is because the flow model parameters θ participate in the calculation of $w_{\text{AIS}}^{(1:L)}$ and $x_{\text{AIS}}^{(1:L)}$.