METROPOLIS-HASTINGS VIEW ON VARIATIONAL INFERENCE AND ADVERSARIAL TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

In this paper we propose to view the acceptance rate of the Metropolis-Hastings algorithm as a universal objective for learning to sample from target distribution – given either as a set of samples or in the form of unnormalized density. This point of view unifies the goals of such approaches as Markov Chain Monte Carlo (MCMC), Generative Adversarial Networks (GANs), variational inference. To reveal the connection we derive the lower bound on the acceptance rate and treat it as the objective for learning explicit and implicit samplers. The form of the lower bound allows for doubly stochastic gradient optimization in case the target distribution factorizes (i.e. over data points). We empirically validate our approach on Bayesian inference for neural networks and generative models for images.

1 INTRODUCTION

Bayesian framework and deep learning have become more and more interrelated during recent years. Recently Bayesian deep neural networks were used for estimating uncertainty (Gal & Ghahramani, 2016), ensembling (Gal & Ghahramani, 2016) and model compression (Molchanov et al., 2017). On the other hand, deep neural networks may be used to improve approximate inference in Bayesian models (Kingma & Welling, 2014).

Learning modern Bayesian neural networks requires inference in the spaces with dimension up to several million by conditioning the weights of DNN on hundreds of thousands of objects. For such applications, one has to perform the approximate inference – predominantly by either sampling from the posterior with Markov Chain Monte Carlo (MCMC) methods or approximating the posterior with variational inference (VI) methods.

MCMC methods are non-parametric, provide the unbiased (in the limit) estimate but require careful hyperparameter tuning especially for big datasets and high dimensional problems. The large dataset problem has been addressed for different MCMC algorithms: stochastic gradient Langevin dynamics (Welling & Teh, 2011), stochastic gradient Hamiltonian Monte Carlo (Chen et al., 2014), minibatch Metropolis-Hastings algorithms (Korattikara et al., 2014; Chen et al., 2016). One way to address the problem of high dimension is the design of a proposal distribution. For example, for the Metropolis-Hastings (MH) algorithm there exists a theoretical guideline for scaling the variance of a Gaussian proposal (Roberts et al., 1997; 2001). More complex proposal designs include adaptive updates of the proposal distribution during iterations of MH algorithm (Holden et al., 2009; Giordani & Kohn, 2010).

Variational inference is extremely scalable but provides a biased estimate of the target distribution. Using the doubly stochastic procedure (Titsias & Lázaro-Gredilla, 2014; Hoffman et al., 2013) VI can be applied to extremely large datasets and high dimensional spaces, such as a space of neural network weights (Kingma et al., 2015; Gal & Ghahramani, 2015; 2016). The bias introduced by variational approximation can be mitigated by using flexible approximations (Rezende & Mohamed, 2015) and resampling (Grover et al., 2018).

Generative Adversarial Networks (Goodfellow et al., 2014) (GANs) is a different approach to learn samplers. Under the framework of adversarial training different optimization problems could be solved efficiently (Arjovsky et al., 2017; Nowozin et al., 2016). The shared goal of "learning to sample" inspired the connection of GANs with VI (Mescheder et al., 2017) and MCMC (Song et al., 2017).

In this paper, we propose a novel perspective on learning to sample from a target distribution by optimizing parameters of either explicit or implicit probabilistic model. Our objective is inspired by the view on the acceptance rate of the Metropolis-Hastings algorithm as a quality measure of the sampler. We derive a lower bound on the acceptance rate and maximize it with respect to parameters of the sampler, treating the sampler as a proposal distribution in the Metropolis-Hastings scheme.

We consider two possible forms of the target distribution: unnormalized density (*density-based* setting) and a set of samples (*sample-based* setting). Each of these settings reveals a unifying property of the proposed perspective and the derived lower bound. In the density-based setting, the lower bound is the sum of forward and reverse KL-divergences between the true posterior and its approximation, connecting our approach to VI. In the sample-based setting, the lower bound admit a form of an adversarial game between the sampler and a discriminator, connecting our approach to GANs.

The closest work to ours is of Song et al. (2017). In contrast to their paper our approach (1) is free from hyperparameters; (2) is able to optimize the acceptance rate directly; (3) avoids minimax problem in the density based setting.

Our main contributions are as follows:

1. We introduce a novel perspective on learning to sample from the target distribution by treating the acceptance rate in the Metropolis-Hastings algorithm as a measure of sampler quality.
2. We derive the lower bound on the acceptance rate allowing for doubly stochastic optimization of the proposal distribution in case when the target distribution factorizes (i.e. over data points).
3. For sample-based and density-based forms of target distribution we show the connection of the proposed algorithm to variational inference and GANs.

The rest of the paper is organized as follows. In Section 2 we introduce the lower bound on the AR. Special forms of target distribution are addressed in Section 3. We validate our approach on the problems of approximate Bayesian inference in the space of high dimensional neural network weights and generative modeling in the space of images in Section 4. We discuss results and directions of the future work in Section 5.

2 Acceptance Rate for Metropolis-Hastings Algorithm

2.1 Preliminaries

In MH algorithm we need to sample from target distribution $p(x)$ while we are only able to sample from proposal distribution $q(x' \mid x)$. One step of the MH algorithm can be described as follows.

1. sample proposal point $x' \sim q(x' \mid x)$, given previously accepted point x
2. accept $\begin{cases} x', & \text{if } \frac{p(x')q(x \mid x')}{p(x)q(x' \mid x)} > u, \quad u \sim \text{Uniform}[0,1] \\ x, & \text{otherwise} \end{cases}$

If the proposal distribution $q(x' \mid x)$ does not depend on x, i.e. $q(x' \mid x) = q(x')$, the algorithm is called *independent* MH algorithm.

The quality of the proposal distribution is measured by acceptance rate and mixing time. Mixing time defines the speed of convergence of the Markov chain to the stationary distribution. The acceptance rate of the MH algorithm is defined as

$$\text{AR} = \mathbb{E}_{\xi} \min\{1, \xi\} = \int dx dx' p(x) q(x' \mid x) \min\left\{1, \frac{p(x')q(x \mid x')}{p(x)q(x' \mid x)}\right\}, \tag{1}$$

where

$$\xi = \frac{p(x')q(x \mid x')}{p(x)q(x' \mid x)}, \quad x \sim p(x), \;\; x' \sim q(x' \mid x). \tag{2}$$

In case of independent proposal distribution we show that the acceptance rate defines a semimetric in distribution space between p and q (see Appendix B).

2.2 OPTIMIZING THE LOWER BOUND ON ACCEPTANCE RATE

Although, we can maximize the acceptance rate of the MH algorithm (Eq. 1) directly w.r.t. parameters ϕ of the proposal distribution $q_\phi(x' \mid x)$, we propose to maximize the lower bound on the acceptance rate. As our experiments show (see Section 4) the optimization of the lower bound compares favorably to the direct optimization of the acceptance rate. To introduce this lower bound we first express the acceptance rate in terms of total variation distance.

Theorem 1 *For random variable* $\xi = \frac{p(x')q(x \mid x')}{p(x)q(x' \mid x)}, x \sim p(x), x' \sim q(x' \mid x)$

$$\mathbb{E}_\xi \min\{1, \xi\} = 1 - \frac{1}{2}\mathbb{E}_\xi|\xi - 1| = 1 - \mathrm{TV}\Big(p(x')q(x \mid x')\Big\|p(x)q(x' \mid x)\Big), \tag{3}$$

where TV *is the total variation distance.*

The proof of Theorem 1 can be found in Appendix A. This reinterpretation in terms of total variation allows us to lower bound the acceptance rate through the Pinsker's inequality

$$\mathrm{AR} \geq 1 - \sqrt{\frac{1}{2} \cdot \mathrm{KL}\Big(p(x')q(x \mid x')\Big\|p(x)q(x' \mid x)\Big)}. \tag{4}$$

The maximization of this lower bound can be equivalently formulated as

$$\mathrm{KL}\Big(p(x')q_\phi(x \mid x')\Big\|p(x)q_\phi(x' \mid x)\Big) \to \min_\phi. \tag{5}$$

In the following sections, we show the benefits of this optimization problem in two different settings — when the target distribution is given in a form of unnormalized density and as a set of samples.

3 OPTIMIZATION OF PROPOSAL DISTRIBUTION

From now on we consider only optimization problem Eq. 5 but the proposed algorithms can be also used for the direct optimization of the acceptance rate (Eq. 1).

To estimate the loss function (Eq. 5) we need to evaluate the density ratio. In the density-based setting unnormalized density of the target distribution is given, so we suggest to use explicit proposal distribution to compute the density ratio explicitly. In the sample-based setting, however, we cannot compute the density ratio, so we propose to approximate it via adversarial training (Goodfellow et al., 2014). The brief summary of constraints for both settings is shown in Table 1.

Table 1: Constraints for two settings of learning sampling algorithms

Setting	**Target distribution**	**Proposal distribution**	**Density Ratio**
Density-based	given $\hat{p}(x) \propto p(x)$	explicit model $q(x')$	explicit
Sample-based	set of samples $X \sim p(x)$	implicit model $q(x')$ implicit model $q(x'\|x)$	learned discriminator

The following subsections describe the algorithms in detail.

3.1 DENSITY-BASED SETTING

In the density-based setting, we assume the proposal to be an explicit probabilistic model, i.e. the model that we can sample from and evaluate its density at any point up to the normalization constant. We also assume that the proposal is reparametrisable (Kingma & Welling, 2014; Rezende et al., 2014; Gal, 2016).

If the proposal belongs to a parametric family, e.g. $q_\phi(x' \mid x) = \mathcal{N}(x' \mid x, \sigma)$ we might face the collapsing to the delta-function problem. To tackle this problem one can properly choose a parametric

family of the proposal, or make the proposal independent $q_\phi(x' \mid x) = q_\phi(x')$. In Appendix C we provide an intuition that shows why the Markov chain proposal can collapse to delta-function and the independent proposal can't. In this section, we consider only the independent proposal. We also provide empirical evidence in section 4 that collapsing to the delta-function does not happen for independent proposal distribution.

Considering $q_\phi(x')$ as the proposal, optimization problem 5 takes the form

$$\mathcal{L}(p, q_\phi) = \mathrm{KL}\Big(p(x')q_\phi(x)\Big\|p(x)q_\phi(x')\Big) = \mathbb{E}_{\substack{x \sim p(x) \\ x' \sim q_\phi(x')}} \log \frac{p(x)q_\phi(x')}{p(x')q_\phi(x)} \to \min_\phi. \tag{6}$$

Explicit form of the proposal $q_\phi(x')$ and the target $p(x)$ distributions allows us to obtain density ratios $q_\phi(x)/q_\phi(x')$ and $p(x')/p(x)$ for any points x, x'. But to estimate the loss in Eq. 6 we also need to obtain samples from the target distribution $x \sim p(x)$ during training. For this purpose, we use the current proposal q_ϕ and run the independent MH algorithm. After obtaining samples from the target distribution it is possible to perform optimization step by taking stochastic gradients w.r.t. ϕ. Pseudo-code for the obtained procedure is shown in Algorithm 1.

Algorithm 1 Optimization of proposal distribution in density-based case

Require: explicit probabilistic model $q_\phi(x')$
Require: density of target distribution $\hat{p}(x) \propto p(x)$
  while ϕ not converged **do**
    sample $\{x'_k\}_{k=1}^K \sim q_\phi(x')$
    sample $\{x_k\}_{k=1}^K \sim p(x)$ using independent MH with current proposal q_ϕ
    $\mathcal{L}(p, q_\phi) \simeq \frac{1}{K}\sum_{k=1}^K \log \frac{p(x_k)q_\phi(x'_k)}{p(x'_k)q_\phi(x_k)}$ ▷ approximate loss with finite number of samples
    $\phi \leftarrow \phi - \alpha \nabla_\phi \mathcal{L}(p, q_\phi)$ ▷ perform gradient descent step
  end while
  return optimal parameters ϕ

Algorithm 1 could also be employed for the direct optimization of the acceptance rate (Eq. 1). Now we apply this algorithm for Bayesian inference problem and show that during optimization of the lower bound we can use minibatches of data, while it is not the case for direct optimization of the acceptance rate. We consider Bayesian inference problem for discriminative model on dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, where x_i is the feature vector of ith object and y_i is its label. For the discriminative model we know likelihood $p(y_i \mid x_i, \theta)$ and prior distribution $p(\theta)$. In order to obtain predictions for some object x_i, we need to evaluate the predictive distribution

$$p(y_i \mid x_i) = \mathbb{E}_{p(\theta \mid \mathcal{D})} p(y_i \mid x_i, \theta). \tag{7}$$

To obtain samples from posterior distribution $p(\theta \mid \mathcal{D})$ we suggest to learn proposal distribution $q_\phi(\theta)$ and perform independent MH algorithm. Thus the optimization problem 6 can be rewritten as follows.

$$\mathcal{L}\Big(p(\theta \mid \mathcal{D}), q_\phi(\theta)\Big) = \mathrm{KL}\Big(p(\theta' \mid \mathcal{D})q_\phi(\theta)\Big\|p(\theta \mid \mathcal{D})q_\phi(\theta')\Big) \to \min_\phi \tag{8}$$

Note that due to the usage of independent proposal, the minimized KL-divergence splits up into the sum of two KL-divergences.

$$\mathrm{KL}\Big(p(\theta' \mid \mathcal{D})q_\phi(\theta)\Big\|p(\theta \mid \mathcal{D})q_\phi(\theta')\Big) = \mathrm{KL}\Big(q_\phi(\theta)\Big\|p(\theta \mid \mathcal{D})\Big) + \mathrm{KL}\Big(p(\theta' \mid \mathcal{D})\Big\|q_\phi(\theta')\Big) \to \min_\phi \tag{9}$$

Minimization of the first KL-divergence corresponds to the variational inference procedure.

$$\mathrm{KL}\Big(q_\phi(\theta)\Big\|p(\theta \mid \mathcal{D})\Big) = -\mathbb{E}_{\theta \sim q_\phi(\theta)} \sum_{i=1}^N \log p(y_i \mid x_i, \theta) + \mathrm{KL}(q_\phi(\theta)\|p(\theta)) + \log p(\mathcal{D}) \tag{10}$$

The second KL-divergence has the only term that depends on ϕ. Thus we obtain the following optimization problem

$$-\mathbb{E}_{\theta \sim q_\phi(\theta)} \sum_{i=1}^N \log p(y_i \mid x_i, \theta) + \mathrm{KL}(q_\phi(\theta)\|p(\theta)) - \mathbb{E}_{\theta \sim p(\theta \mid \mathcal{D})} \log q_\phi(\theta) \to \min_\phi. \tag{11}$$

The first summand here contains the sum over all objects in dataset $\mathcal{D}$. We follow doubly stochastic variational inference and suggest to perform unbiased estimation of the gradient in Eq. 11 using only minibatches of data. Moreover, we can use recently proposed techniques (Korattikara et al., 2014; Chen et al., 2016) that perform the independent MH algorithm using only minibatches of data. Combination of these two techniques allows us to use only minibatches of data during iterations of algorithm 1. In the case of the direct optimization of the acceptance rate, straightforward usage of minibatches results in biased gradients. Indeed, for the direct optimization of the acceptance rate (Eq. 1) we have the product over the all training data inside $\min$ function.

3.2 Sample-based Setting

In the sample-based setting, we assume the proposal to be an implicit probabilistic model, i.e. the model that we can only sample from. As in the density-based setting, we assume that we are able to perform the reparameterization trick for the proposal.

In this subsection we consider only Markov chain proposal $q_\phi(x' \,|\, x)$, but everything can be applied to independent proposal $q_\phi(x')$ by simple substitution $q_\phi(x' \,|\, x)$ with $q_\phi(x')$. From now we will assume our proposal distribution to be a neural network that takes x as its input and outputs x'. Considering proposal distribution parameterized by a neural network allows us to easily exclude delta-function from the space of solutions. We avoid learning the identity mapping by using neural networks with the bottleneck and noisy layers. For the detailed description of the architectures see Appendix E.

The set of samples from the true distribution $X \sim p(x)$ allows for the Monte Carlo estimation of the loss

$$\mathcal{L}(p, q_\phi) = \mathbb{E}_{\substack{x \sim p(x) \\ x' \sim q_\phi(x' \,|\, x)}} \log \frac{p(x) q_\phi(x' \,|\, x)}{p(x') q_\phi(x \,|\, x')} \to \min_\phi . \tag{12}$$

To compute the density ratio $\frac{p(x) q_\phi(x' \,|\, x)}{p(x') q_\phi(x \,|\, x')}$ we suggest to use well-known technique of density ratio estimation via training discriminator network. Denoting discriminator output as $D(x, x')$, we suggest the following optimization problem for the discriminator.

$$-\mathbb{E}_{\substack{x \sim p(x) \\ x' \sim q_\phi(x' \,|\, x)}} \log D(x, x') - \mathbb{E}_{\substack{x \sim p(x) \\ x' \sim q_\phi(x' \,|\, x)}} \log(1 - D(x', x)) \to \min_D \tag{13}$$

Speaking informally, such discriminator takes two images as input and tries to figure out which image is sampled from true distribution and which one is generated by the one step of proposal distribution. It is easy to show that optimal discriminator in problem 13 will be

$$D(x, x') = \frac{p(x) q_\phi(x' \,|\, x)}{p(x) q_\phi(x' \,|\, x) + p(x') q_\phi(x \,|\, x')}. \tag{14}$$

Note that for optimal discriminator we have $D(x, x') = 1 - D(x', x)$. In practice, we have no optimal discriminator and these values can differ significantly. Thus, we have four ways for density ratio estimation that may differ significantly.

$$\frac{p(x) q_\phi(x' \,|\, x)}{p(x') q_\phi(x \,|\, x')} \approx \frac{D(x, x')}{1 - D(x, x')} \approx \frac{1 - D(x', x)}{D(x', x)} \approx \frac{1 - D(x', x)}{1 - D(x, x')} \approx \frac{D(x, x')}{D(x', x)} \tag{15}$$

To avoid the ambiguity we suggest to use the discriminator of a special structure. Let $\widetilde{D}(x, x')$ be a convolutional neural network with scalar output. Then the output of discriminator $D(x, x')$ is defined as follows.

$$D(x, x') = \frac{\exp(\widetilde{D}(x, x'))}{\exp(\widetilde{D}(x, x')) + \exp(\widetilde{D}(x', x))} \tag{16}$$

In other words, such discriminator can be described as the following procedure. For single neural network $\widetilde{D}(\cdot, \cdot)$ we evaluate two outputs $\widetilde{D}(x, x')$ and $\widetilde{D}(x', x)$. Then we take softmax operation for these values. Summing up all the steps, we obtain algorithm 2.

Algorithm 2 Optimization of proposal distribution in sample-based case

Require: implicit probabilistic model $q_\phi(x' \mid x)$
Require: large set of samples $X \sim p(x)$
for n iterations **do**
  sample $\{x_k\}_{k=1}^K \sim X$
  sample $\{x'_k\}_{k=1}^K \sim q_\phi(x'|x)$
  train discriminator D by optimizing 13
  $\mathcal{L}(p, q_\phi) \approx \frac{1}{K}\sum_{k=1}^K \log \frac{D(x_k, x'_k)}{1-D(x_k, x'_k)}$ ▷ approximate loss with finite number of samples
  $\phi \leftarrow \phi - \alpha \nabla_\phi \mathcal{L}(p, q_\phi)$ ▷ perform gradient descent step
end for
return parameters ϕ

Algorithm 2 could also be employed for direct optimization of the acceptance rate (Eq. 1). But, in Appendix F we provide an intuition for this setting that the direct optimization of the acceptance rate may struggle from vanishing gradients.

4 Experiments

In this section, we provide experiments for both density-based and sample-based settings, showing the proposed procedure is applicable to high dimensional target distributions. Code for reproducing all of the experiments will be published with the camera-ready version of the paper.

4.1 Toy Problem

This experiment shows that it is possible to optimize the acceptance rate, optimizing its lower bound. For the target distribution we consider bimodal Gaussian $p(x) = 0.5 \cdot \mathcal{N}(x \mid -2, 0.5) + 0.5 \cdot \mathcal{N}(x \mid 2, 0.7)$, for the independent proposal we consider unimodal gaussian $q(x) = \mathcal{N}(x \mid \mu, \sigma)$. We perform stochastic gradient optimization from the same initialization for both objectives (Fig. 1) and obtain approximately the same local maximums.

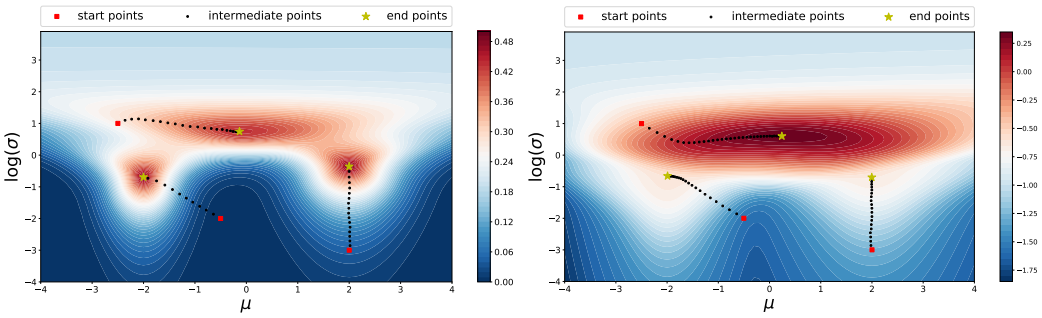


Figure 1: Level-plots in parameter space for the toy problem. Left: level-plot for the acceptance rate of the MH algorithm. Right: level-plot for the lower bound of the acceptance rate.

4.2 Density-based Setting

In density-based setting, we consider Bayesian inference problem for the weights of a neural network. In our experiments we consider approximation of predictive distribution (Eq. 7) as our main goal. To estimate the goodness of the approximation we measure negative log-likelihood and accuracy on the test set.

In subsection 3.1 we show that lower bound on acceptance rate can be optimized more efficiently than acceptance rate due to the usage of minibatches. But other questions arise.

1. Does the proposed objective in Eq. 11 allow for better estimation of predictive distribution compared to the variational inference?
2. Does the application of the MH correction to the learned proposal distribution allow for better estimation of the predictive distribution (Eq. 7) than estimation via raw samples from the proposal?

To answer these questions we consider reduced LeNet-5 architecture (see Appendix D) for classification task on 20k images from MNIST dataset (for test data we use all of the MNIST test set). Even after architecture reduction we still face a challenging task of learning a complex distribution in 8550-dimensional space. For the proposal distribution we use fully-factorized gaussian $q_\phi(\theta) = \prod_{j=1}^{d} \mathcal{N}(\theta_j \,|\, \mu_j, \sigma_j)$ and standard normal distribution for prior $p(\theta) = \prod_{j=1}^{d} \mathcal{N}(\theta_j \,|\, 0, 1)$.

For variational inference, we train the model using different initialization and pick the model according to the best ELBO. For our procedure, we do the same and choose the model by the maximum value of the acceptance rate lower bound. In algorithm 1 we propose to sample from the posterior distribution using the independent MH and the current proposal. It turns out in practice that it is better to use the currently learned proposal $q_\phi(\theta) = \mathcal{N}(\theta \,|\, \boldsymbol{\mu}, \boldsymbol{\sigma})$ as the initial state for random-walk MH algorithm. That is, we start with the mean $\boldsymbol{\mu}$ as an initial point, and then use random-walk proposal $q(\theta' \,|\, \theta) = \mathcal{N}(\theta' \,|\, \theta, \boldsymbol{\sigma})$ with the variances $\boldsymbol{\sigma}$ of current independent proposal. This should be considered as a heuristic that improves the approximation of the loss function.

The optimization of the acceptance rate lower bound results in the better estimation of predictive distribution than the variational inference (see Fig. 2). Optimization of acceptance rate for the same number of epochs results in nearly 30% accuracy on the test set. That is why we do not report results for this procedure in Fig. 2.

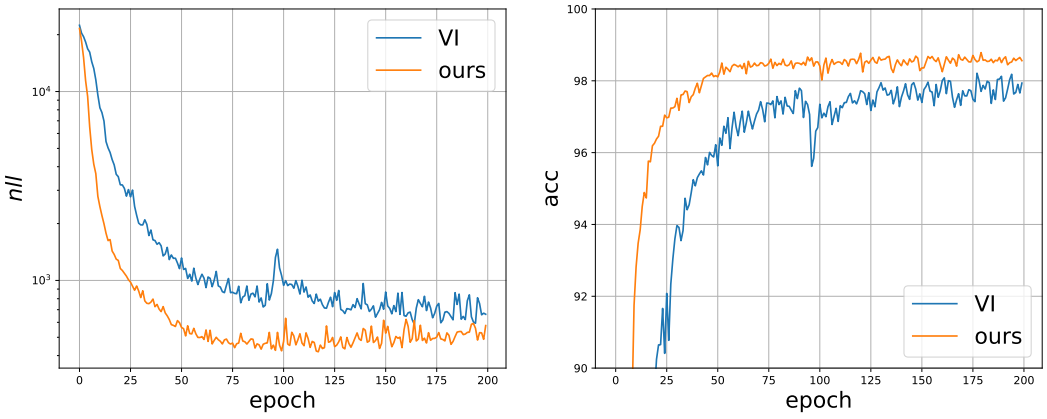


Figure 2: Negative log-likelihood (left) and accuracy (right) on test set of MNIST dataset for variational inference (blue lines) and the optimization of the acceptance rate lower bound (orange lines). In both procedures we apply the independent MH algorithm to estimate the predictive distribution.

To answer the second question we estimate predictive distribution in two ways. The first way is to perform 100 accept/reject steps of the independent MH algorithm with the learned proposal $q_\phi(\theta)$ after each epoch, i.e. perform MH correction of the samples from the proposal. The second way is to take the same number of samples from $q_\phi(\theta)$ without MH correction. For both estimations of predictive distribution, we evaluate negative log-likelihood on the test set and compare them.

The MH correction of the learned proposal improves the estimation of predictive distribution for the variational inference (right plot of Fig. 3) but does not do so for the optimization of the acceptance rate lower bound (left plot of Fig. 3). This fact may be considered as an implicit evidence that our procedure learns the proposal distribution with higher acceptance rate.

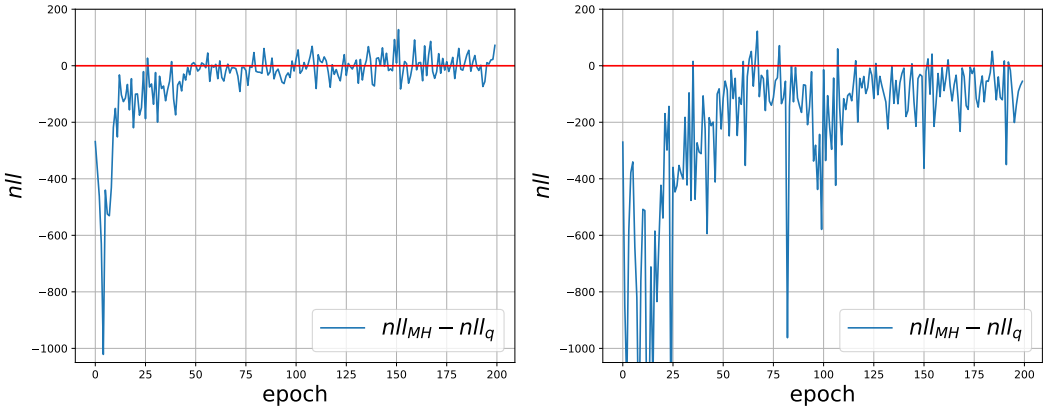


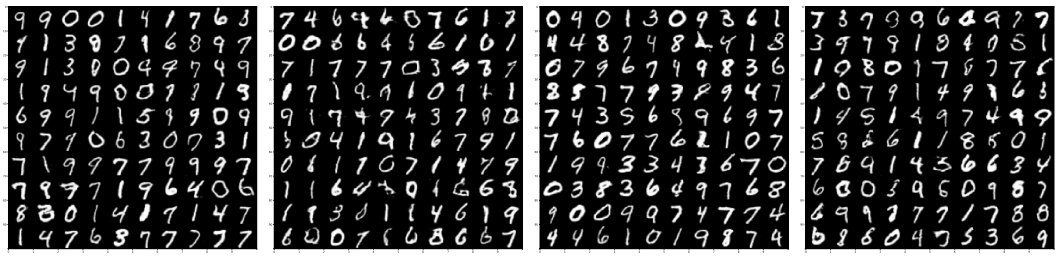
Figure 3: Test negative log-likelihood for two approximations of the predictive distribution based on samples: from proposal distribution nll_q and after MH correction nll_{MH}. Left figure corresponds to the optimization of the acceptance rate lower bound, right figure corresponds to the variational inference.

4.3 Sample-based Setting

In the sample-based setting, we estimate density ratio using a discriminator. Hence we do not use the minibatching property (see subsection 3.1) of the obtained lower bound, and optimization problems for the acceptance rate and for the lower bound have the same efficiency in terms of using data. That is why our main goal in this setting is to compare the optimization of the acceptance rate and the optimization of the lower bound. Also, in this setting, we have Markov chain proposal that is interesting to compare with the independent proposal. Summing up, we formulate the following questions:

1. Does the optimization of the lower bound has any benefits compared to the direct optimization of the acceptance rate?
2. Do we have mixing issue while learning Markov chain proposal in practice?
3. Could we improve the visual quality of samples by applying the MH correction to the learned proposal?

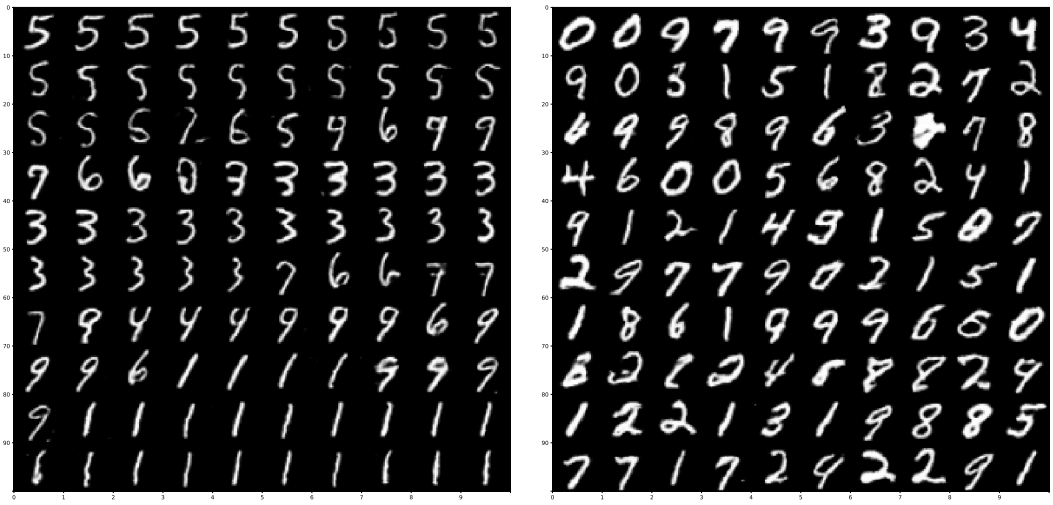
We use DCGAN architecture for the proposal and discriminator (see Appendix E) and apply our algorithm to the MNIST dataset. We consider two optimization problems: direct optimization of the acceptance rate and its lower bound. We also consider two ways to obtain samples from the approximation of the target distribution — use raw samples from the learned proposal, or perform the MH algorithm, where we use the learned discriminator for density ratio estimation.



(a) (b) (c) (d)

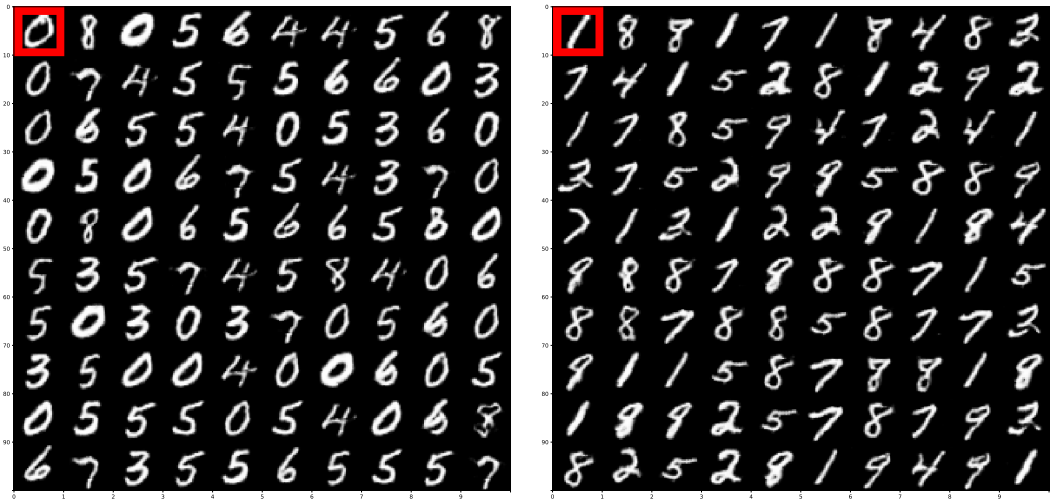
Figure 4: Samples from the learned independent proposal obtained via optimization: of acceptance rate (4(a), 4(b)) and its lower bound (4(c), 4(d)). In Fig. 4(b), 4(d) we show raw samples from the learned proposal. In Fig. 4(a), 4(c) we show the samples after applying the independent MH correction to the samples, using the learned discriminator for density ratio estimation.

In case of the independent proposal, we show that the MH correction at evaluation step allows to improve visual quality of samples — figures 4(a) and 4(b) for the direct optimization of acceptance rate, figures 4(c) and 4(d) for the optimization of its lower bound. Note that in Algorithm 2 we do not apply the independent MH algorithm during training. Potentially, one can use the MH algorithm considering any generative model as a proposal distribution and learning a discriminator for density ratio estimation. Also, for this proposal, we demonstrate the negligible difference in visual quality of samples obtained by the direct optimization of acceptance rate (see Fig. 4(a)) and by the optimization of the lower bound (see Fig. 4(c)).



(a) (b)

Figure 5: Samples from the chain obtained via the MH algorithm with the learned proposal and the learned discriminator for density ratio estimation. Fig. 5(a) corresponds to the direct optimization of the acceptance rate. Fig. 5(b) – to optimization of the lower bound on acceptance rate. Samples in the chain are obtained one by one from left to right from top to bottom.



(a) (b)

Figure 6: Samples from the proposal distribution and conditioned on the digit in the red box. The proposal was optimized according to the lower bound on the acceptance rate.

In the case of the Markov chain proposal, we show that the direct optimization of acceptance rate results in slow mixing (see Fig. 5(a)) — most of the time the proposal generates samples from one of the modes (digits) and rarely switches to another mode. When we perform the optimization of the lower bound the proposal switches between modes frequently (see Fig. 5(b)).

To show that the learned proposal distribution has the Markov property rather than being totally independent, we show samples from the proposal conditioned on two different points in the dataset (see Fig. 6). The difference in samples from two these distributions (Fig. 6(a), 6(a)) reflects the dependence on the conditioning.

Additionally, in Appendix G we present samples from the chain after 10000 accepted images and also samples from the chain that was initialized with noise.

5 Discussion and future work

This paper proposes to use the acceptance rate of the MH algorithm as the universal objective for learning to sample from some target distribution. We also propose the lower bound on the acceptance rate that should be preferred over the direct maximization of the acceptance rate in many cases. The proposed approach provides many ways of improvement by the combination with techniques from the recent developments in the field of MCMC, GANs, variational inference. For example

- The quality of a sampler in density-based setting could be improved with the normalizing flows (Rezende & Mohamed, 2015).
- We can use stochastic Hamiltonian Monte Carlo (Chen et al., 2014) for the loss estimation in Algorithm 1.
- In sample-based setting one can use more advanced techniques of density ratio estimation.

Another interesting direction of further research is the design of the family of explicit Markov chain proposals resistant to the collapsing to the delta-function problem. Application of the MH algorithm to improve the quality of generative models also requires exhaustive further exploration and rigorous treatment.

References

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Haoyu Chen, Daniel Seita, Xinlei Pan, and John Canny. An efficient minibatch acceptance test for metropolis-hastings. *arXiv preprint arXiv:1610.06848*, 2016.

Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *International Conference on Machine Learning*, pp. 1683–1691, 2014.

Yarin Gal. Uncertainty in deep learning. *PhD thesis, University of Cambridge*, 2016.

Yarin Gal and Zoubin Ghahramani. Bayesian convolutional neural networks with bernoulli approximate variational inference. *arXiv preprint arXiv:1506.02158*, 2015.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059, 2016.

Paolo Giordani and Robert Kohn. Adaptive independent metropolis–hastings by fast estimation of mixtures of normals. *Journal of Computational and Graphical Statistics*, 19(2):243–259, 2010.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Aditya Grover, Ramki Gummadi, Miguel Lazaro-Gredilla, Dale Schuurmans, and Stefano Ermon. Variational rejection sampling. In Amos Storkey and Fernando Perez-Cruz (eds.), *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pp. 823–832, Playa Blanca, Lanzarote, Canary Islands, 09–11 Apr 2018. PMLR. URL `http://proceedings.mlr.press/v84/grover18a.html`.

Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

Lars Holden, Ragnar Hauge, Marit Holden, et al. Adaptive independent metropolis–hastings. *The Annals of Applied Probability*, 19(1):395–413, 2009.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *ICLR*, 2014.

Diederik P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems*, pp. 2575–2583, 2015.

Anoop Korattikara, Yutian Chen, and Max Welling. Austerity in mcmc land: Cutting the metropolis-hastings budget. In *International Conference on Machine Learning*, pp. 181–189, 2014.

Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. *arXiv preprint arXiv:1701.04722*, 2017.

Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. Variational dropout sparsifies deep neural networks. *arXiv preprint arXiv:1701.05369*, 2017.

Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in Neural Information Processing Systems*, pp. 271–279, 2016.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *ICML*, 2014.

Gareth O Roberts, Andrew Gelman, Walter R Gilks, et al. Weak convergence and optimal scaling of random walk metropolis algorithms. *The annals of applied probability*, 7(1):110–120, 1997.

Gareth O Roberts, Jeffrey S Rosenthal, et al. Optimal scaling for various metropolis-hastings algorithms. *Statistical science*, 16(4):351–367, 2001.

Jiaming Song, Shengjia Zhao, and Stefano Ermon. A-nice-mc: Adversarial training for mcmc. In *Advances in Neural Information Processing Systems*, pp. 5140–5150, 2017.

Michalis Titsias and Miguel Lázaro-Gredilla. Doubly stochastic variational bayes for non-conjugate inference. In *International Conference on Machine Learning*, pp. 1971–1979, 2014.

Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 681–688, 2011.

A PROOF OF THEOREM 1

Remind that we have random variables $\xi = \frac{p(x')q(x\,|\,x')}{p(x)q(x'\,|\,x)}, x \sim p(x), x' \sim q(x'\,|\,x)$ and $u \sim \text{Uniform}[0,1]$, and want to prove the following equalities.

$$\mathbb{E}_\xi \min\{1,\xi\} = \mathbb{P}\{\xi > u\} = 1 - \frac{1}{2}\mathbb{E}_\xi|\xi - 1| \tag{17}$$

Equality $\mathbb{E}_\xi \min\{1,\xi\} = \mathbb{P}\{\xi > u\}$ is obvious.

$$\mathbb{E}_\xi \min\{1,\xi\} = \int_0^\infty p_\xi(x)\min\{1,x\}dx = \int_{x\geq 1} p_\xi(x)dx + \int_{x<1} p_\xi(x)xdx \tag{18}$$

$$\mathbb{P}\{\xi > u\} = \int_0^\infty dx p_\xi(x)\int_0^x [0 \leq u \leq 1]du = \int_{x\geq 1} p_\xi(x)dx + \int_{x<1} p_\xi(x)xdx \tag{19}$$

Equality $\mathbb{P}\{\xi > u\} = 1 - \frac{1}{2}\mathbb{E}_\xi|\xi - 1|$ can be proofed as follows.

$$\mathbb{P}\{\xi > u\} = \int_0^1 du \int_u^{+\infty} p_\xi(x)dx = \int_0^1 (1 - F_\xi(u))du = \tag{20}$$

$$= 1 - \left[uF_\xi(u)\Big|_0^1 - \int_0^1 up_\xi(u)du\right] = 1 - F_\xi(1) + \int_0^1 up_\xi(u)du, \tag{21}$$

where $F_\xi(u)$ is CDF of random variable ξ. Note that $F_\xi(0) = 0$ since $\xi \in (0, +\infty]$. Eq. 21 can be rewritten in two ways.

$$1 - F_\xi(1) + \int_0^1 up_\xi(u)du = 1 + \int_0^1 (u-1)p_\xi(u)du = 1 - \int_0^1 |u-1|p_\xi(u)du \tag{22}$$

To rewrite Eq. 21 in the second way we note that $\mathbb{E}\xi = 1$.

$$1 - F_\xi(1) + \int_0^1 up_\xi(u)du = \int_1^{+\infty} p_\xi(u)du + 1 - \int_1^{+\infty} up_\xi(u)du = 1 - \int_1^{+\infty} |u-1|p_\xi(u)du \tag{23}$$

Summing equations 22 and 23 results in the following formula

$$\mathbb{P}\{\xi > u\} = 1 - \frac{1}{2}\mathbb{E}_\xi|\xi - 1|. \tag{24}$$

Using the form of ξ we can rewrite the acceptance rate as

$$1 - \frac{1}{2}\mathbb{E}_\xi|\xi - 1| = 1 - \text{TV}\Big(p(x')q(x\,|\,x')\Big\|p(x)q(x'\,|\,x)\Big). \tag{25}$$

B ACCEPTANCE RATE OF INDEPENDENT MH DEFINES SEMIMETRIC IN DISTRIBUTION SPACE

In independent case we have $\xi = \frac{p(x')q(x)}{p(x)q(x')}, x \sim p(x), x' \sim q(x')$ and we want to prove that $\mathbb{E}_\xi|\xi - 1|$ is semimetric (or pseudo-metric) in space of distributions. For this appendix, we denote $D(p,q) = \mathbb{E}_\xi|\xi - 1|$. The first two axioms for metric obviously holds

1. $D(p,q) = 0 \iff p = q$
2. $D(p,q) = D(q,p)$

There is an example when triangle inequality does not hold. For distributions $p = \text{Uniform}[0, 2/3], q = \text{Uniform}[1/3, 1], s = \text{Uniform}[0,1]$

$$D(p,s) + D(q,s) = \frac{4}{3} < \frac{3}{2} = D(p,q). \tag{26}$$

But weaker inequality can be proved.

$$D(p,s) + D(q,s) = \int |p(x)s(y) - p(y)s(x)|dydx + \int |q(x)s(y) - q(y)s(x)|dydx = \tag{27}$$

$$= \int \Big[|\underbrace{p(x)s(y)q(z)}_{a} - \underbrace{p(y)s(x)q(z)}_{b}| + |\underbrace{q(x)s(y)p(z)}_{c} - \underbrace{q(y)s(x)p(z)}_{d}| \Big] dxdydz \tag{28}$$

$$D(p,s) + D(q,s) = \int |p(z)s(y)q(x) - p(y)s(z)q(x)|dxdydz + \tag{29}$$

$$+ \int |q(x)s(z)p(y) - q(z)s(x)p(y)|dxdydz \geq \int \Big| \underbrace{q(x)s(y)p(z)}_{c} - \underbrace{p(y)s(x)q(z)}_{b} \Big| dxdydz \tag{30}$$

$$D(p,s) + D(q,s) = \int |p(z)s(x)q(y) - p(x)s(z)q(y)|dxdydz + \tag{31}$$

$$+ \int |q(y)s(z)p(x) - q(z)s(y)p(x)|dxdydz \geq \int \Big| \underbrace{q(y)s(x)p(z)}_{d} - \underbrace{p(x)s(y)q(z)}_{a} \Big| dxdydz \tag{32}$$

Summing up equations 28, 30 and 32 we obtain

$$3(D(p,s) + D(q,s)) \geq \int dxdydz \Big[|a-b| + |c-d| + |c-b| + |d-a| \Big] \geq 2 \int dxdydz |d-b| = \tag{33}$$

$$= 2 \int dxdydz s(x) \Big| q(y)p(z) - q(z)p(y) \Big| = 2D(p,q) \tag{34}$$

$$D(p,s) + D(q,s) \geq \frac{2}{3} D(p,q) \tag{35}$$

C ON COLLAPSING TO THE DELTA-FUNCTION

Firstly, let's consider the case of gaussian random-walk proposal $q(x' \,|\, x) = \mathcal{N}(x' \,|\, x, \sigma)$. The optimization problem for the acceptance rate takes the form

$$\text{AR} = \int dxdx' p(x) \mathcal{N}(x' \,|\, x, \sigma) \min \left\{ 1, \frac{p(x')}{p(x)} \right\} \to \max_{\sigma}. \tag{36}$$

It is easy to see that we can obtain acceptance rate arbitrarly close to 1, taking σ small enough.

In the case of the independent proposal, we don't have the collapsing to the delta-function problem. In our work, it is important to show non-collapsing during optimization of the lower bound, but the same hold for the direct optimization of the acceptance rate. To provide such intuition we consider one-dimensional case where we have some target distribution $p(x)$ and independent proposal $q(x) = \mathcal{N}(x \,|\, \mu, \sigma)$. Choosing σ small enough, we approximate sampling with the independent MH as sampling on some finite support $x \in [\mu - a, \mu + a]$. For this support, we approximate the target distribution with the uniform distribution (see Fig. 7).

For such approximation, optimization of lower bound takes the form

$$\text{KL}(p(x) \| q(x)) + \text{KL}(q(x) \| p(x)) \to \min_{q} \tag{37}$$

$$\text{KL}(\text{Uniform}[-a,a] \| \mathcal{N}(x \,|\, 0, \sigma, -a, a)) + \text{KL}(\mathcal{N}(x \,|\, 0, \sigma, -a, a) \| \text{Uniform}[-a,a]) \to \min_{\sigma} \tag{38}$$

Here $\mathcal{N}(x \,|\, 0, \sigma, -a, a)$ is truncated normal distribution. The first KL-divergence can be written as follows.

$$\text{KL}(\text{Uniform}[-a,a] \| \mathcal{N}(x \,|\, 0, \sigma, -a, a)) = -\frac{1}{2a} \int_{-a}^{a} dx \log \mathcal{N}(x \,|\, 0, \sigma, -a, a) - \log 2a = \tag{39}$$

$$= -\frac{1}{2a} \Big[-2a \log(\sigma Z) - a \log 2\pi - \frac{1}{2\sigma^2} \frac{2a^3}{3} \Big] - \log 2a = \tag{40}$$

$$= \log \sigma + \log Z + \frac{a^2}{6\sigma^2} + \frac{1}{2} \log 2\pi - \log 2a \tag{41}$$

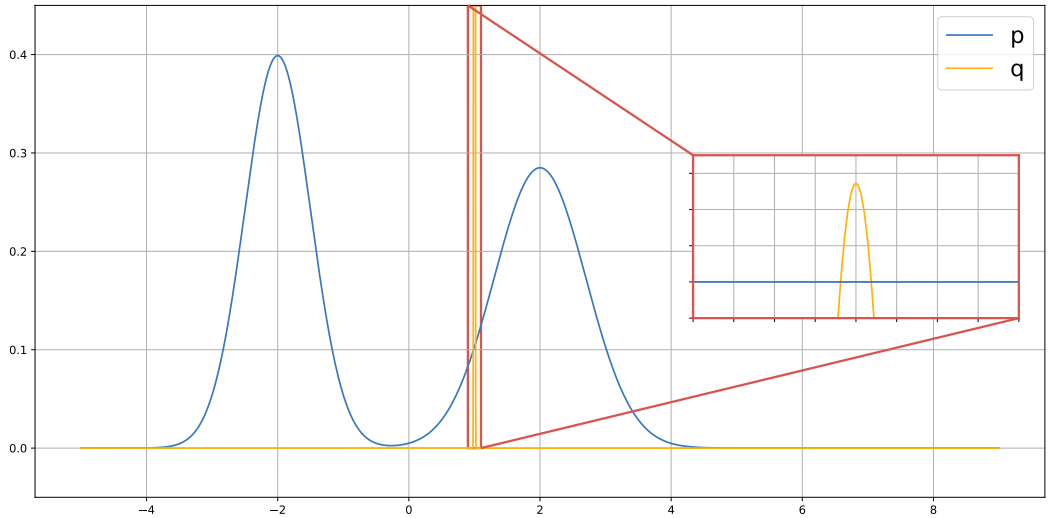


Figure 7: In this figure we show schematic view of approximation of of target distribution with uniform distribution. Red bounding box is made bigger for better comprehension.

Here Z is normalization constant of truncated log normal distribution and $Z = \Phi(a/\sigma) - \Phi(-a/\sigma)$, where $\Phi(x)$ is CDF of standard normal distribution. The second KL-divergence is

$$\mathrm{KL}(\mathcal{N}(x \,|\, 0, \sigma, -a, a) \| \mathrm{Uniform}[-a, a]) = \tag{42}$$

$$= -\frac{1}{2}\log(2\pi e) - \log\sigma - \log Z + \frac{a}{\sqrt{2\pi}\sigma Z}\exp\left(-\frac{a^2}{2\sigma^2}\right) + \log 2a \tag{43}$$

Summing up two KL-divergencies and taking derivative w.r.t. σ we obtain

$$\frac{\partial}{\partial\sigma}\Big(\mathrm{KL}(\mathrm{Uniform}[-a, a] \| \mathcal{N}(x \,|\, 0, \sigma, -a, a)) + \mathrm{KL}(\mathcal{N}(x \,|\, 0, \sigma, -a, a) \| \mathrm{Uniform}[-a, a])\Big) = \tag{44}$$

$$= -\frac{a^2}{3\sigma^3} + \frac{a^3}{\sqrt{2\pi}\sigma^4 Z}\exp\left(-\frac{a^2}{2\sigma^2}\right) + \frac{a}{\sqrt{2\pi}}\exp\left(-\frac{a^2}{2\sigma^2}\right)\left[-\frac{1}{\sigma^2 Z} - \frac{1}{\sigma Z^2}\frac{-2a}{\sigma^2\sqrt{2\pi}}\exp\left(-\frac{a^2}{2\sigma^2}\right)\right] = \tag{45}$$

$$= \frac{1}{a}\left[-\frac{a^3}{3\sigma^3} + \frac{a^2}{\sqrt{2\pi}\sigma^2 Z}\exp\left(-\frac{a^2}{2\sigma^2}\right)\left(\frac{a^2}{\sigma^2} - 1 + \frac{2a}{\sqrt{2\pi}\sigma Z}\exp\left(-\frac{a^2}{2\sigma^2}\right)\right)\right] \tag{46}$$

To show that the derivative of the lower bound w.r.t. σ is negative, we need to prove that the following inequality holds for positive x.

$$-\frac{1}{3}x^3 + \frac{x^2}{\sqrt{2\pi}(\Phi(x) - \Phi(-x))}\exp(-x^2/2)\left(x^2 - 1 + \frac{2x}{\sqrt{2\pi}(\Phi(x) - \Phi(-x))}\exp(-x^2/2)\right) < 0, \quad x > 0 \tag{47}$$

Defining $\phi(x) = \int_0^x e^{-t^2/2}dt$ and noting that $2\phi(x) = \sqrt{2\pi}(\Phi(x) - \Phi(-x))$ we can rewrite inequality 47 as

$$\frac{1}{\phi(x)}e^{-x^2/2}\left(x^2 - 1 + \frac{2xe^{-x^2/2}}{\phi(x)}\right) < \frac{2x}{3}, \quad x > 0 \tag{48}$$

By the fundamental theorem of calculus, we have

$$xe^{-x^2/2} = \int_0^x e^{-t^2/2}(1 - t^2)dt \tag{49}$$

Hence,

$$\phi(x) - xe^{-x^2/2} = \int_0^x e^{-t^2/2}t^2 dt \geq e^{-x^2/2}\int_0^x t^2 dt = e^{-x^2/2}\frac{x^3}{3} \tag{50}$$

Or equivalently,

$$\phi(x) \geq e^{-x^2/2}\frac{x^3+3x}{3} \tag{51}$$

Using this inequality twice, we obtain

$$\frac{e^{-x^2/2}}{\phi(x)} \leq \frac{3}{x(x^2+3)} \tag{52}$$

and

$$x^2 - 1 + \frac{xe^{-x^2/2}}{\phi(x)} \leq x^2 - 1 + \frac{3}{x^2+3} = \frac{x^2(2+x^2)}{x^2+3} \tag{53}$$

Thus, the target inequality can be verified by the verification of

$$\frac{3x(2+x^2)}{(x^2+3)^2} \leq \frac{2x}{3}. \tag{54}$$

Thus, we show that partial derivative of our lower bound w.r.t. σ is negative. Using that knowledge we can improve our loss by taking a bigger value of σ. Hence, such proposal does not collapse to delta-function.

D ARCHITECTURE OF THE REDUCED LENET-5

```
class LeNet5(BayesNet):
    def __init__(self):
        super(LeNet5, self).__init__()
        self.num_classes = 10
        self.conv1 = layers.ConvFFG(1, 10, 5, padding=0)
        self.relu1 = nn.ReLU(True)
        self.pool1 = nn.MaxPool2d(2, padding=0)
        self.conv2 = layers.ConvFFG(10, 20, 5, padding=0)
        self.relu2 = nn.ReLU(True)
        self.pool2 = nn.MaxPool2d(2, padding=0)
        self.flatten = layers.ViewLayer([20*4*4])
        self.dense1 = layers.LinearFFG(20*4*4, 10)
        self.relu3 = nn.ReLU()
        self.dense2 = layers.LinearFFG(10, 10)
```

E ARCHITECTURES OF NEURAL NETWORKS IN SAMPLE-BASED SETTING

In sample-based setting we use usual DCGAN architecture for independent proposal distribution

```
class Generator(layers.ModuleWrapper):
    def __init__(self):
        super(Generator, self).__init__()
        self.fc = nn.Linear(100, 128*8*8)
        self.unflatten = layers.ViewLayer([128, 8, 8])
        self.in1 = nn.InstanceNorm2d(128)
        self.us1 = nn.ConvTranspose2d(128, 128, 2, 2)
        self.conv1 = nn.Conv2d(128, 128, 3, stride=1, padding=1)
        self.in2 = nn.InstanceNorm2d(128, 0.8)
        self.lrelu1 = nn.LeakyReLU(0.2, inplace=True)
        self.us2 = nn.ConvTranspose2d(128, 128, 2, 2)
        self.conv2 = nn.Conv2d(128, 64, 3, stride=1, padding=1)
        self.in3 = nn.InstanceNorm2d(64, 0.8)
        self.lrelu2 = nn.LeakyReLU(0.2, inplace=True)
        self.conv3 = nn.Conv2d(64, 1, 3, stride=1, padding=1)
        self.tanh = nn.Tanh()
```

And a little be modified acrhitecture for Markov chain proposal distribution

```
class Generator(layers.ModuleWrapper):
    def __init__(self):
        super(Generator, self).__init__()

        self.d_conv1 = nn.Conv2d(1, 16, 5, stride=2, padding=2)
        self.d_lrelu1 = nn.LeakyReLU(0.2, inplace=True)
        self.d_do1 = nn.Dropout2d(0.5)
        self.d_conv2 = nn.Conv2d(16, 4, 5, stride=2, padding=2)
        self.d_in2 = nn.InstanceNorm2d(4, 0.8)
        self.d_lrelu2 = nn.LeakyReLU(0.2, inplace=True)
        self.d_do2 = nn.Dropout2d(0.5)

        self.b_view = layers.ViewLayer([4*8*8])
        self.b_fc = nn.Linear(4*8*8, 256)
        self.b_lrelu = nn.LeakyReLU(0.2, inplace=True)
        self.b_fc = nn.Linear(256, 128 * 8 * 8)
        self.b_do = layers.AdditiveNoise(0.5)

        self.e_unflatten = layers.ViewLayer([128, 8, 8])
        self.e_in1 = nn.InstanceNorm2d(128, 0.8)
        self.e_us1 = nn.ConvTranspose2d(128, 128, 2, 2)
        self.e_conv1 = nn.Conv2d(128, 128, 3, stride=1, padding=1)
        self.e_in2 = nn.InstanceNorm2d(128, 0.8)
        self.e_lrelu1 = nn.LeakyReLU(0.2, inplace=True)
        self.e_us2 = nn.ConvTranspose2d(128, 128, 2, 2)
        self.e_conv2 = nn.Conv2d(128, 64, 3, stride=1, padding=1)
        self.e_in3 = nn.InstanceNorm2d(64, 0.8)
        self.e_lrelu2 = nn.LeakyReLU(0.2, inplace=True)
        self.e_conv3 = nn.Conv2d(64, 1, 3, stride=1, padding=1)
        self.e_tanh = nn.Tanh()
```

For both proposals we use the proposed discriminator with the following architecture.

```
class Discriminator(nn.Module):
    def __init__(self):
        super(Discriminator, self).__init__()
        self.conv1 = nn.Conv2d(2, 16, 3, 2, 1)
        self.lrelu1 = nn.LeakyReLU(0.2, inplace=True)
        self.conv2 = nn.Conv2d(16, 32, 3, 2, 1)
        self.lrelu2 = nn.LeakyReLU(0.2, inplace=True)
        self.in2 = nn.InstanceNorm2d(32, 0.8)
        self.conv3 = nn.Conv2d(32, 64, 3, 2, 1)
        self.lrelu3 = nn.LeakyReLU(0.2, inplace=True)
        self.in3 = nn.InstanceNorm2d(64, 0.8)
        self.conv4 = nn.Conv2d(64, 128, 3, 2, 1)
        self.lrelu4 = nn.LeakyReLU(0.2, inplace=True)
        self.in4 = nn.InstanceNorm2d(128, 0.8)
        self.flatten = layers.ViewLayer([128*2*2])
        self.fc = nn.Linear(128*2*2, 1)

    def forward(self, x, y):
        xy = torch.cat([x, y], dim=1)
        for module in self.children():
            xy = module(xy)
        yx = torch.cat([y, x], dim=1)
        for module in self.children():
            yx = module(yx)
        return F.softmax(torch.cat([xy, yx], dim=1), dim=1)
```

F INTUITION FOR BETTER GRADIENTS IN SAMPLE-BASED SETTING

In this section, we provide an intuition for sample-based setting that the loss function for lower bound has better gradients than the loss function for acceptance rate. Firstly, we remind that in the sample-based setting we use a discriminator for density ratio estimation.

$$D(x, x') = \frac{p(x)q(x' \mid x)}{p(x)q(x' \mid x) + p(x')q(x \mid x')} \tag{55}$$

For this purpose we use the discriminator of special structure

$$D(x, x') = \frac{\exp(\widetilde{D}(x, x'))}{\exp(\widetilde{D}(x, x')) + \exp(\widetilde{D}(x', x))} = \frac{1}{1 + \exp\Big(- (\widetilde{D}(x, x') - \widetilde{D}(x', x))\Big)} \tag{56}$$

We denote $d(x, x') = \widetilde{D}(x, x') - \widetilde{D}(x', x)$ and consider the case when the discriminator can easily distinguish fake pairs from valid pairs. So $D(x, x')$ is close to 1 and $d(x, x') \gg 0$ for $x \sim p(x)$ and $x' \sim q(x' \mid x)$. To evaluate gradients we consider Monte Carlo estimations of each loss and take gradients w.r.t. x' in order to obtain gradients for parameters of proposal distribution. We do not introduce the reparameterization trick to simplify the notation but assume it to be performed. For the optimization of the acceptance rate we have

$$\int dx dx' p(x)q(x' \mid x) \left| \frac{p(x')q(x \mid x')}{p(x)q(x' \mid x)} - 1 \right| \simeq \left| \frac{p(x')q(x \mid x')}{p(x)q(x' \mid x)} - 1 \right| \tag{57}$$

$$L_{\text{AR}} = \left| \frac{p(x')q(x \mid x')}{p(x)q(x' \mid x)} - 1 \right| \approx \left| \frac{1 - D(x, x')}{D(x, x')} - 1 \right| \tag{58}$$

$$\frac{\partial L_{\text{AR}}}{\partial x'} = \frac{1}{D^2(x, x')} \frac{\partial D(x, x')}{\partial x'} = \exp(-d(x, x')) \frac{\partial d(x, x')}{\partial x'} \tag{59}$$

While for the optimization of the lower bound we have

$$\int dx dx' p(x)q(x' \mid x) \log \left(\frac{p(x)q(x' \mid x)}{p(x')q(x \mid x')} \right) \simeq \log \left(\frac{p(x)q(x' \mid x)}{p(x')q(x \mid x')} \right) \tag{60}$$

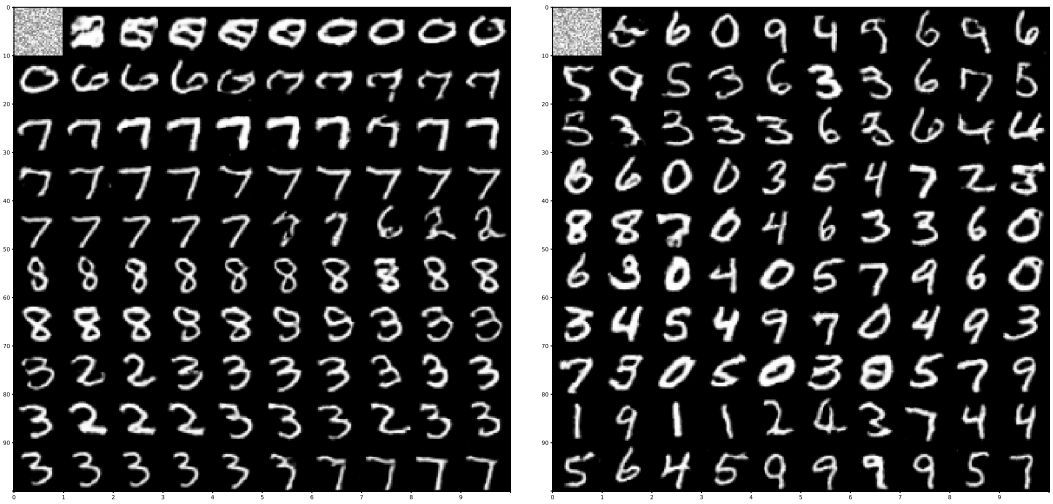
$$L_{\text{LB}} = -\log \left(\frac{p(x')q(x \mid x')}{p(x)q(x' \mid x)} \right) \approx -\log \left(\frac{1 - D(x, x')}{D(x, x')} \right) \tag{61}$$

$$\frac{\partial L_{\text{LB}}}{\partial x'} = \frac{1}{(1 - D(x, x'))D(x, x')} \frac{\partial D(x, x')}{\partial x'} = \frac{\partial d(x, x')}{\partial x'} \tag{62}$$

Now we compare Eq. 59 and Eq. 62. We see that in case of strong discriminator we have vanishing gradients in Eq. 59 due to $\exp(-d(x, x'))$, while it is not the case for Eq. 62.

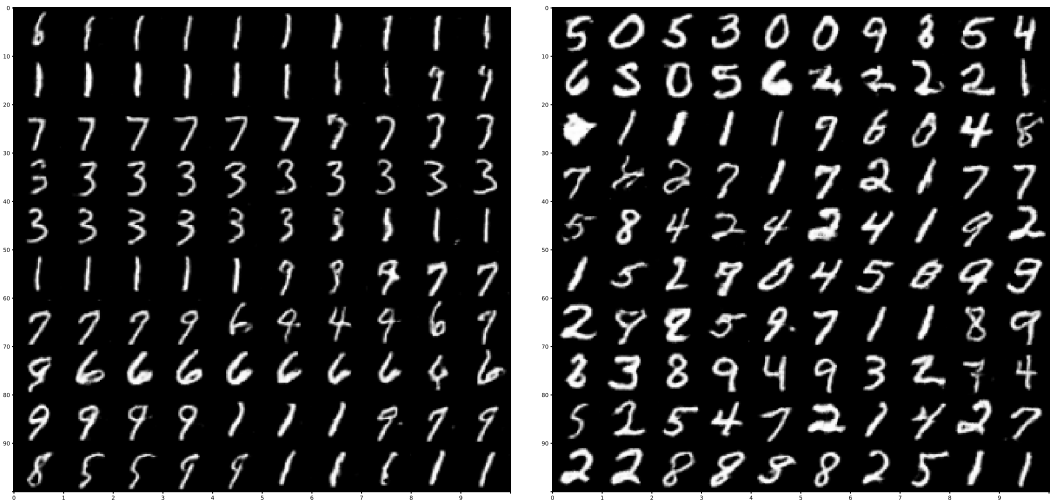
G ADDITIONAL FIGURES FOR MARKOV CHAIN PROPOSALS IN SAMPLE-BASED SETTING

In this section, we show additional figures for Markov chain proposals. In Fig. 8 we show samples from the chain that was initialized by the noise. In Fig. 9 we show samples from the chain after 10000 accepted samples.



(a) (b)

Figure 8: Samples from the chain initialized with noise. To obtain samples we use the MH algorithm with the learned proposal and the learned discriminator for density ratio estimation. In Fig. 5(a) we use proposal and discriminator that are learned during optimization of acceptance rate. In Fig. 5(b) we use proposal and discriminator that are learned during the optimization of the acceptance rate lower bound. Samples in the chain are obtained one by one from left to right from top to bottom starting with noise (first image in the figure).



(a) (b)

Figure 9: Samples from the chain after 10000 accepted samples. To obtain samples we use the MH algorithm with the learned proposal and the learned discriminator for density ratio estimation. In Fig. 5(a) we use proposal and discriminator that are learned during optimization of acceptance rate. In Fig. 5(b) we use proposal and discriminator that are learned during the optimization of the acceptance rate lower bound. Samples in chain are obtained one by one from left to right from top to bottom.