UNIVERSITY OF CAMBRIDGE

Improved Ergodic Inference via Kernelised Stein Discrepancy

Wenlong Chen

Darwin College

This dissertation is submitted on August 19, 2020 for the degree of Master of Philosophy in Machine Learning and Machine Intelligence

Declaration

I, Wenlong Chen of Darwin College, being a candidate for the MPhil in Machine Learning and Machine Intelligence, hereby declare that this report and the work described in it are my own work, unaided except as may be specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose. This dissertation contains 12559 words.

The implementation of the proposed model was written in Python. The code was adapted from the vanilla Ergodic Inference codes provided by Dr José Miguel Hernández-Lobato and Dr Yichuan Zhang. The implementation of KSD was adapted from Dr Yingzhen Li's code and the implementation of maxSKSD was adapted from Wenbo Gong's code.

Wenlong Chen
August, 2020

Acknowledgements

I would like to thank my supervisor Dr José Miguel Hernández-Lobato for advising me on this dissertation. Thank you for providing interesting ideas to explore and nudging me in the right directions all the time.

I would also like to thank Wenbo Gong for providing useful suggestions on the implementation of the models. I have had a great deal to learn from them.

Abstract

Markov Chain Monte Carlo (MCMC) and Variational Inference (VI) are both popular statistical inference methods in machine learning. MCMC is guaranteed to draw unbiased samples from target distribution asymptotically but the samples are correlated and it often requires long burn-in simulations, which makes it computationally expensive. VI, based on optimisation, is more computationally efficient and we can draw independent samples from the parametric approximation of the target distribution, but VI can perform poorly when the target distribution is complicated and the parametric approximation is not able to represent it well. Ergodic Inference (EI) is a hybrid method that combines MCMC and VI to balance between computational cost and approximation bias. EI generates samples from approximate distribution by running multiple Hamilton Monte Carlo (HMC) MCMC chains, with initial distribution found by VI, for a fixed number of iterations. Then the last states of these chains are independent approximate posterior samples. However, EI tends to suffer from the mode collapse pathology because the hyperparameters of HMC are tuned by optimizing a new objective resulting from ignoring the entropy term in the evidence lower bound used by VI. The mode collapse pathology can be detrimental to the performance of EI. However, it was found that if the initial distribution has high entropy, the pathology can be avoided. In this work, we mitigate EI's mode collapse pathology by tuning an inflation parameter and then using it to scale the variational distribution found by VI to increase the variance of the initial distribution. The inflation parameter will be determined using Kernelized Stein Discrepancy (KSD) or max Sliced Kernelized Stein Discrepancy (maxSKSD).

Contents

List of Figures

List of Tables

Chapter 1

Background

In this chapter, we introduce the problem to deal with in this project and review some previous methods that our proposed model is based on.

In probabilistic models, the distributions we would like to learn are most likely to be very hard to work with analytically. As a concrete example, the posterior distribution of parameters θ given dataset $\mathcal{D}$ and predictive distribution over new data point $\mathbf{x}^*$ in Bayesian Inference [1] are

$$P(\theta|\mathcal{D}) = \frac{P(\mathcal{D}|\theta)P(\theta)}{P(\mathcal{D})} \tag{1.1}$$

$$P(\mathbf{y}^*|\mathbf{x}^*, \mathcal{D}) = \int P(\mathbf{y}^*|\mathbf{x}^*, \theta)P(\theta|\mathcal{D})d\theta \tag{1.2}$$

For almost all practical Bayesian Inference problems, the evidence (denominator of (1.1)) $P(\mathcal{D})$ is intractable since it requires us to compute a complex integral with no analytical solution and since the dimension of θ is usually very high, numerical integration is also infeasible due to high computational cost. Consequently, the posterior distribution is intractable and we have to use approximate inference methods.

In many complex models of interest, we have a target distribution $\pi(\mathbf{x})$ which is impractical to work with analytically and it is also hard to draw samples from it directly. Moreover, like in Bayesian Inference problem described above, the normalization constant $\mathcal{Z}$ of the target distribution is often unknown, therefore, we can only work with an unnormalized target $\pi^*(\mathbf{x})$. In machine learning literature, Variational Inference (VI) [2] and Markov Chain Monte Carlo (MCMC) [3] are two most popular statistical inference methods to address this problem.

1.1 Variational Inference

Variational Inference (VI) based on optimization aims at finding the member from a family of parametric distributions, such as factorized Gaussian, that is most similar to the target distribution in the sense that the Kullback-Leibler divergence (KL-divergence) between this variational distribution and the target distribution is minimized. KL-divergence is always non-negative with 0 being achieved if and only if the two distributions are the same, and KL-divergence is an asymmetric measure between two distributions ($\mathcal{D}_{KL}(p||\pi) \neq \mathcal{D}_{KL}(\pi||p)$ in general).

Suppose we only consider using distribution from a family of parametric distributions $p(\mathbf{x};\theta)$ to approximate the target distribution $\pi(\mathbf{x})$ and we only know $\pi(\mathbf{x})$ up to a normalisation constant $\mathcal{Z}$, $\pi^*(\mathbf{x}) = \mathcal{Z}\pi(\mathbf{x})$, where θ is the variational distribution parameters, then our goal is to find the optimal variational distribution parameters θ^* such that the KL-divergence between $p(\mathbf{x};\theta)$ and $\pi(\mathbf{x})$, $\mathcal{D}_{KL}(p(\mathbf{x};\theta)||\pi(\mathbf{x}))$ is minimized. Fortunately, $\mathcal{D}_{KL}(p(\mathbf{x};\theta)||\pi(\mathbf{x}))$ can be minimized indirectly by maximizing the evidence lower bound even without knowing the normalization constant:

$$\begin{aligned}
\theta^* &= \operatorname*{argmin}_{\theta} \mathcal{D}_{KL}(p(\mathbf{x};\theta)||\pi(\mathbf{x})) \\
&= \operatorname*{argmin}_{\theta} E_p[\log \frac{p(\mathbf{x};\theta)}{\pi(\mathbf{x})}] \\
&= \operatorname*{argmin}_{\theta} E_p[\log \frac{p(\mathbf{x};\theta)}{\pi^*(\mathbf{x})}] - \log \mathcal{Z} \\
&= \operatorname*{argmax}_{\theta} E_p[\log \pi^*(\mathbf{x}) - \log p(\mathbf{x};\theta)]
\end{aligned} \tag{1.3}$$

The objective to maximize in the last line of (1.3) is the evidence lower bound L_{ELBO}, and it is easy to show that $L_{ELBO} = \log \mathcal{Z} - \mathcal{D}_{KL}(p(\mathbf{x};\theta)||\pi(\mathbf{x}))$. Since KL-divergence is always non-negative, L_{ELBO} is a lower bound of $\log \mathcal{Z}$. In Bayesian Inference, $\mathcal{Z} = P(\mathcal{D})$, which is the evidence or marginal likelihood of the data. Maximizing L_{ELBO} can be viewed as a way to indirectly maximize the marginal likelihood of the data. However, the marginal likelihood of the data is not guaranteed to increase every time when L_{ELBO} increases. It is worth noting that L_{ELBO} can be decomposed into two terms: the first term is expected logarithm of the unnormalized target distribution with respect to $p(\mathbf{x};\theta)$, $E_p[\log \pi^*(\mathbf{x})]$, and the second term is the entropy of $p(\mathbf{x};\theta)$, $H(p) = -E_p[\log(p(\mathbf{x};\theta))]$. The first term should be reasonably high if $p(\mathbf{x};\theta)$ matches π well since the samples generated from $p(\mathbf{x};\theta)$ should fall in high probability mass region of $\pi(\mathbf{x})$ frequently if the variational distribution is similar to the target. However, maximizing only the first term can result in mode collapse problem. In an extreme case, if p is a fully flexible distribution, then the variational distribution found after optimization would be a Dirac-Delta distribution at the mode of the target π. The second term prevents p from shrinking to the degenerate delta solution since maximizing the entropy encourages a distribution with more uncertainty.

If the chosen family of distributions contains the true target distribution π, then the variational approximation is unbiased ($\mathcal{D}_{KL}(p(\mathbf{x};\theta)||\pi(\mathbf{x})) = 0$). However, in practice, to ensure L_{ELBO} is tractable, we often can only work with some simple families of distributions such as factorized Gaussian to ensure the second term ($H(p)$) in L_{ELBO} is tractable. Consequently, the family of distributions chosen is rarely rich enough to capture the target for practical models and therefore the approximation bias is almost inevitable when VI is used. When the expressiveness of the chosen family of distributions is not capable of representing the target well, the approximation bias of VI can be large and it may considerably hurt the performance of the model. While VI usually can only give us biased estimates, the fact that it is based on optimization significantly reduces the computational cost compared with methods based on sampling like Markov Chain Monte Carlo, which we will discuss in next section. It is also worth noting that we can often generate independent samples from the variational distribution when we are required to compute Monte Carlo estimate based on the variational distribution, since the parametric family of distributions is often simple enough for us to draw independent samples directly.

1.2 Markov Chain Monte Carlo and Hamiltonian Monte Carlo

Markov Chain Monte Carlo (MCMC) is a simulation approach that builds a Markov Chain with the target distribution $\pi(\mathbf{x})$ as its stationary distribution by repeatedly applying an appropriate transition operator $\mathbf{x}_t \sim T(\mathbf{x}_t|\mathbf{x}_{t-1})$ that satisfies the Detailed Balance (DB) condition: $\pi(\mathbf{x})T(\mathbf{x}'|\mathbf{x}) = \pi(\mathbf{x}')T(\mathbf{x}|\mathbf{x}')$. Typically, MCMC techniques achieve it by first sampling an auxiliary random variable $\mathbf{r}$ from an auxiliary distribution q_θ with parameters θ, then constructing a new augmented samples $(\mathbf{x}', \mathbf{r}') = f_\phi(\mathbf{x}_{t-1}, \mathbf{r})$ through some proper deterministic function f_ϕ with parameters ϕ and finally including a Metropolis-Hasting correction step: $\mathbf{x}_t = \mathbf{x}'$ with probability $p = \min\{0, \frac{\pi(\mathbf{x}')q_\theta(\mathbf{r}')}{\pi(\mathbf{x}_{t-1})q_\theta(\mathbf{r})}\}$, otherwise set $\mathbf{x}_t = \mathbf{x}_{t-1}$. Note that the Metropolis-Hasting step does not require us to know the normalization constant of π since $\frac{\pi(\mathbf{x}')}{\pi(\mathbf{x}_{t-1})} = \frac{\pi^*(\mathbf{x}')}{\pi^*(\mathbf{x}_{t-1})}$.

Traditional MCMC methods like Metropolis-Hasting algorithm is inefficient when dealing with complex correlated high-dimensional target distribution because we have to use transition operator that only proposes states close to the current state to avoid falling in low target probability regions by taking a large random step. It results in highly correlated samples and prevents the Markov chain to fully explore the target distribution. We can overcome this inefficiency by using Hamiltonian Monte Carlo [4], which utilizes the gradient information of log-target to allow the sampler to more efficiently explore the state space.

Hamiltonian Monte Carlo (HMC) is one of the most successful approaches in the MCMC family. It can draw unbiased samples from the target distribution asymptotically without knowing its normalization constant. This property is desirable because we often only have an unnormalized target distribution $\pi^*(\mathbf{x})$ in complex probabilistic models. HMC introduces auxiliary momentum variables r with same dimensionality as the space of the state $\mathbf{x}$ and it considers sampling as simulating time-evolution of a fictitious physical system whose dynamics can explore the state space.

Let the augmented space containing state and momentum be the phase space $\mathbf{z} = (\mathbf{x}, \mathbf{r})$ and then we simulate the differential equation with respect to $\frac{d\mathbf{z}}{dt}$ as follows:

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \frac{\partial H}{\partial \mathbf{r}} \\ \frac{d\mathbf{r}}{dt} &= -\frac{\partial H}{\partial \mathbf{x}} \end{aligned} \tag{1.4}$$

where $H(\mathbf{x}, \mathbf{r}) = U(\mathbf{x}) + K(\mathbf{r})$, $U(\mathbf{x}) = -\log \pi^*(\mathbf{x})$ is the potential energy, which is equal to the negative logarithm of the unnormalized target, $K(r) = \frac{1}{2}\mathbf{r}^{\mathbf{T}}\mathbf{M}^{-\mathbf{1}}\mathbf{r}$ is called kinetic energy and $\mathbf{M}$ is the mass matrix. In practice, we have to approximate the continuous time system by discretising the differential equations. Thus, we used the leapfrog integrator with discretisation step size hyperparameter and we implemented the vanilla leapfrog integrator following [5] in this work:

Algorithm 1 Leapfrog

```
Input: $\mathbf{x}$: state, $\mathbf{r}$: momentum, $\phi_1$: step size, $\phi_2$: $\mathbf{r}$ variance, $m$: number of steps
Output: $\mathbf{x}'$: new state, $\mathbf{r}'$: new momentum
 1: $\mathbf{x}^* = \mathbf{x}$
 2: $\mathbf{r}^* = \mathbf{r}$
 3: for t in 1 to m do
 4:     $\mathbf{r}^* = \mathbf{r}^* - \frac{1}{2}\phi_1 \frac{\partial}{\partial \mathbf{x}} U(\mathbf{x}^*)$
 5:     $\mathbf{x}^* = \mathbf{x}^* + \frac{\phi_1}{\phi_2}\mathbf{r}^*$
 6:     $\mathbf{r}^* = \mathbf{r}^* - \frac{1}{2}\phi_1 \frac{\partial}{\partial \mathbf{x}} U(\mathbf{x}^*)$
 7: end for
 8: $\mathbf{x}' = \mathbf{x}^*$
 9: $\mathbf{r}' = \mathbf{r}^*$
10: return $\mathbf{x}'$, $\mathbf{r}'$
```

One can see from the leapfrog algorithm shown above that we only need to know the unnormalized target $\pi^*(\mathbf{x})$ to take advantage of the gradient information of log-target since the score function of the target (gradient of log-target) can be computed without knowing the normalization constant: $\nabla_{\mathbf{x}} \log \pi(\mathbf{x}) = \frac{\nabla_{\mathbf{x}} \pi(\mathbf{x})}{\pi(\mathbf{x})} = \frac{\nabla_{\mathbf{x}} \pi^*(\mathbf{x})}{\pi^*(\mathbf{x})} = \nabla_{\mathbf{x}} \log \pi^*(\mathbf{x})$. If we can simulate the differential equations perfectly, then the Hamiltonian H should be constant ([4]). However, since leapfrog integrator is a discrete approximation of the differential equations, it does not preserve energy exactly and we need to incorporate a Metropolis-Hasting correction step at the end of leapfrog to ensure the stationary distribution of the Markov Chain is indeed the target: we accept the new state ($\mathbf{z}' = (\mathbf{x}', \mathbf{r}')$) with probability $\min(1, e^{-H(\mathbf{x},\mathbf{r})+H(\mathbf{x}',\mathbf{r}')})$. If the simulation of the differential equations is very accurate then the acceptance probability will be close to 1. HMC with length L repeats the leapfrog integrator for L iterations and at each iteration, we resample the momentum to ensure sufficient exploration of the target. The step sizes ϕ_1 and momentum variances ϕ_2 in different iterations can be different.

Although HMC is more efficient than many traditional MCMC techniques by using auxiliary momentum variables and simulating Hamiltonian dynamics to allow the particle to move longer distances in the target surface, it is still much more computational demanding compared with VI, especially when dealing with complex high dimensional distributions because it may take impractically many iterations to converge. Another problem of HMC is that like other MCMC techniques, the samples generated by HMC are correlated, which may result in high variance of the Monte Carlo estimator ([5]). Moreover, careful tuning of the hyperparameters of HMC is critical for it to work well in practice. The step sizes should have proper values to balance between insufficient exploration (too small step sizes) and instability (too large step sizes).

1.3 Hamiltonian Ergodic Inference and the mode collapse pathology

In previous two sections, we introduced two popular approximate inference methods VI and HMC. We also discussed their drawbacks and the potential causes of the failure of these two techniques. In short, approximation bias is almost inherent in VI and MCMC is computationally demanding. In this section, we introduce a hybrid method Ergodic Inference (EI), proposed in [6], that combines VI and MCMC to balance between computational cost and approximation bias. More specifically, we focus on Ergodic Inference that combines VI and HMC, which is called Hamiltonian Ergodic Inference (HEI). HEI runs multiple finite length HMC chains with T HMC iterations and finally returns the last states of theses chains, which form independent samples that can be used to approximate the target distribution $\pi(\mathbf{x})$. The initial samples of HEI are drawn from the member of a family of parametric distributions $q_\psi(\mathbf{x})$ whose parameters are tuned by maximizing L_{ELBO} and we use $P_0(\mathbf{x}; \psi)$ to denote this initial variational distribution.

Furthermore, the HMC hyperparameters θ like step sizes and momentum variances are also tuned using ideas borrowed from VI: since we do not have an analytic form for the distribution of the final state of HMC (P_T), the entropy term in L_{ELBO} is intractable and we can not tune the HMC hyperparameters θ by maximizing L_{ELBO}. Instead, we tune the HMC hyperparameters θ by maximizing a new objective resulting from dropping the intractable entropy term from L_{ELBO}, which is the expected logarithm of the unnormalized target term described in section 1.1. The training scheme described above can be summarized as follows:

$$\begin{aligned} &\max_{\theta} E_{P_T(\mathbf{x};\theta,\psi)}[\log \pi^*(\mathbf{x})] \\ &\max_{\psi} E_{P_0(\mathbf{x};\psi)}[\log \pi^*(\mathbf{x}) - \log P_0(\mathbf{x};\psi)] \end{aligned} \tag{1.5}$$

where $\pi^*(\mathbf{x})$ is the unnormalized target distribution, and from now on we use L_{EI} to denote $E_{P_T(\mathbf{x};\theta,\psi)}[\log \pi^*(\mathbf{x})]$.

As discussed in section 1.1, tuning HMC hyperparameters by maximizing L_{EI} alone can lead to mode collapse pathology and it was found in [6] that it is necessary for the initial distribution from which the initial states are sampled to have reasonably large entropy. More specifically, they proposed that the entropy of the initial distribution should be greater than the entropy of the target distribution ($H(P_0) > H(\pi)$) because it will force $E_{P_0}[\log \pi^*(\mathbf{x})]$ to be less than $E_\pi[\log \pi^*(\mathbf{x})]$: since we have $\log \mathcal{Z} = E_\pi[\log \pi^*(\mathbf{x})] + H(\pi) \geq L_{ELBO} = E_{P_0}[\log \pi^*(\mathbf{x})] + H(P_0)$, then $E_\pi[\log \pi^*(\mathbf{x})] - E_{P_0}[\log \pi^*(\mathbf{x})] \geq H(P_0) - H(\pi)$. Thus, if $H(P_0) > H(\pi)$, we have $E_\pi[\log \pi^*(\mathbf{x})] > E_{P_0}[\log \pi^*(\mathbf{x})]$. It is more likely to avoid the mode collapse pathology during maximization of L_{EI} with $E_\pi[\log \pi^*(\mathbf{x})] > E_{P_0}[\log \pi^*(\mathbf{x})]$, than with $E_\pi[\log \pi^*(\mathbf{x})] < E_{P_0}[\log \pi^*(\mathbf{x})]$, because we would expect the maximization of L_{EI} will push up $E_{P_T}[\log \pi^*(\mathbf{x})]$ from $E_{P_0}[\log \pi^*(\mathbf{x})]$.

Unfortunately, variational approximation of π, found by L_{ELBO} maximization, tends to underestimate the uncertainty ([7]) and the entropy of this initial variational distribution is usually not large enough to avoid the mode collapse pathology. We demonstrate it by reproducing the toy experiment of using HEI to generate samples from a correlated bi-variate Gaussian distribution from [6]. The target 2D Gaussian distribution is $\mathcal{N}(x; \mathbf{0}, [[2.0, 1.5], [1.5, 1.6]])$. We use HMC with $T = 30$ iterations (each iteration contains $m = 5$ leapfrog steps) and we consider different step sizes and momentum variances for each HMC iteration and for each sample dimension. We examine the results of using three different initial distributions: (a) P_0 is a factorized Gaussian with entropy less than the entropy of the target, $P_0 \sim \mathcal{N}(\mathbf{x}; \mathbf{0}, 0.5^2\mathbf{I})$, (b) P_0 is a factorized Gaussian with entropy greater than the entropy of the target, $P_0 \sim \mathcal{N}(\mathbf{x}; \mathbf{0}, 3^2\mathbf{I})$ and (c) P_0 is the variational mean field approximation of $\pi(\mathbf{x})$ tuned by maximizing L_{ELBO}. The histograms of samples generated by HEI with these three different initial distributions are plotted in Figure 1.1 and Table 1.1 shows the expected negative log-target estimated using the final states generated by HEI, $-E_{P_{30}}[\log \pi(\mathbf{x})]$. The ground-truth is estimated by independent samples directly drawn from the Gaussian target.

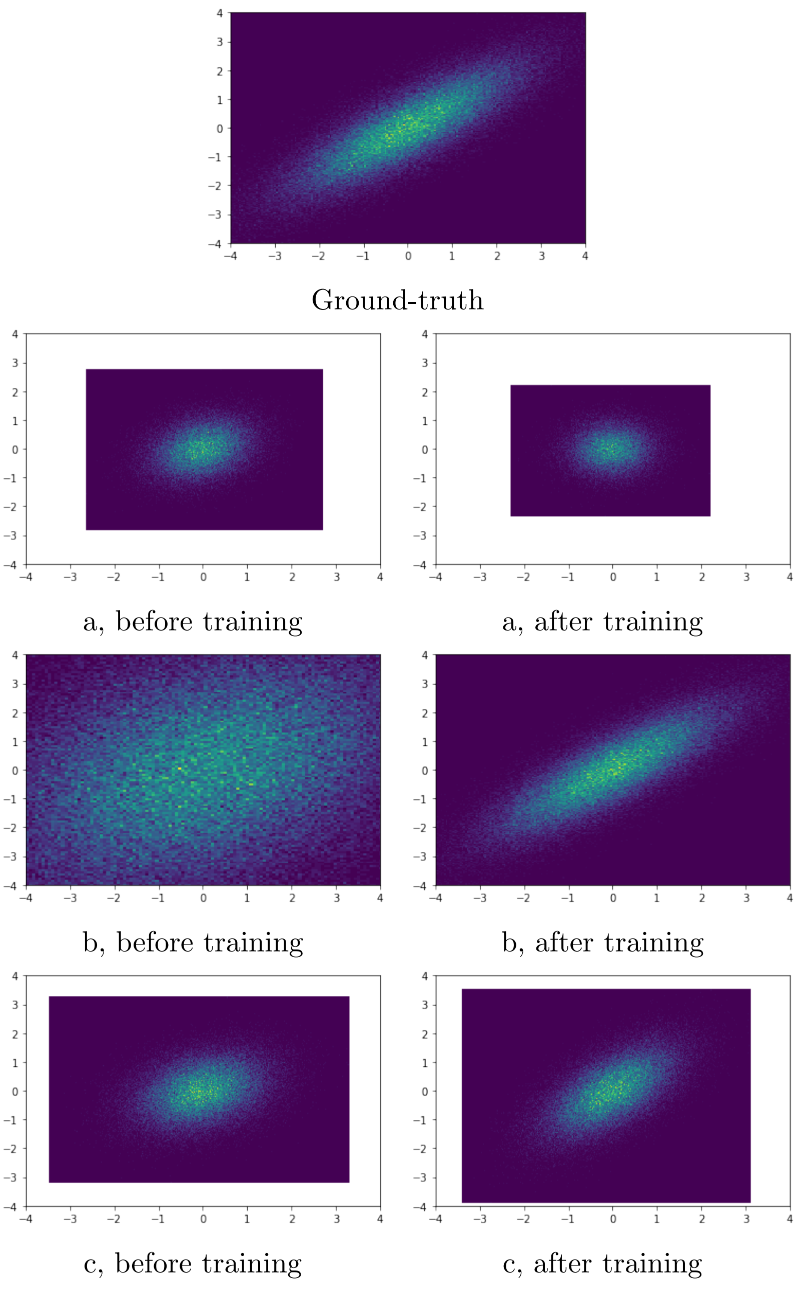


Figure 1.1: Histograms of samples generated by trained HEI or untrained HEI with different initial distributions

Figure 1.1 and Table 1.1 shows that HEI using initial distribution with small entropy (a, c) tends to generate samples close to the mode and the mode collapse pathology is more serious after optimizing L_{EI} with respect to the HMC hyperparameters. The

Ground-truth: $-E_\pi[\log \pi(\mathbf{x})] = 2.8083$			
	a	b	c
Before training	2.3563	9.8907	2.6000
After training	2.2948	**2.8176**	2.5089

Table 1.1: Expected negative log-target estimated using the final states generated by trained HEI or untrained HEI, $-E_{P_{30}}[\log \pi(\mathbf{x})]$

entropy of the variational approximation of the target, found by L_{ELBO} maximization, is also insufficient to avoid the pathology. While HEI using initial distribution with large entropy (b) can result in overestimated uncertainty and unstable performance before training of the HMC hyperparameters, it gives best convergence after training, suggesting reasonably large entropy of the initial distribution is indeed crucial for HEI to work well.

As demonstrated by the toy Gaussian example, the variational approximation found by L_{ELBO} maximization tends to have insufficient entropy for HEI to work well. Our goal in this project is to improve the performance of HEI by finding an initial distribution with sufficient entropy to avoid the mode collapse pathology. We propose to achieve it using two techniques. The first one is to find the initial distribution by optimizing another objective, α-divergence [8], instead of L_{ELBO}. Optimizing L_{ELBO} is equivalent to optimizing α-divergence with $\alpha \to 0$ and with a lager α, such as $\alpha = 1$, the initial distribution found by α-divergence minimization with large α tends to have lager width and it might help overcome the pathology. The second method is to multiply the marginal variances of the initial distribution found by optimizing L_{ELBO} or α-divergence by a reasonably large inflation value s. We also want a method to automatically tune s so that P_T can match π well. In this project, we try to achieve this goal by tuning s with Kernelized Stein Discrepancy (KSD) or a modified version of KSD, called maxSKSD. We will discuss these two techniques in detail in next chapter.

1.4 Related Work

Similar to HEI, there have been many attempts to combine VI and MCMC in recent years to balance between approximation bias and computational cost. As discussed previously, the intractable entropy term in L_{ELBO} prevents us from treating the family of distributions of the final state of HMC $p(\mathbf{x_T})$ as a variational family in practice. [9] proposed to tackle the intractability of $H(p(\mathbf{x_T}))$ by explicitly constructing a lower bound L_{aux} of L_{ELBO} and then optimize L_{aux} directly. L_{aux} can be computed as an expectation with respect to the joint distribution of all intermediate states of HMC by subtracting non-negative expected KL-divergence between $p(\mathbf{x}_{0:T-1}|\mathbf{x}_T)$ and $r(\mathbf{x}_{0:T-1}|\mathbf{x}_T)$ with respect to $p(\mathbf{x}_T)$ from L_{ELBO}, where $r(\mathbf{x}_{0:T-1}|\mathbf{x}_T)$ is an auxiliary reverse model which approximates the reverse dynamics

$p(\mathbf{x}_{0:T-1}|\mathbf{x}_T)$:

$$\begin{aligned} L_{aux} &= E_{p(\mathbf{x}_{0:T})}[\log \pi^*(\mathbf{x}_T) + \log r(\mathbf{x}_{0:T-1}) - \log p(\mathbf{x}_{0:T})] \\ &= E_{p(\mathbf{x}_T)}[\log \pi^*(\mathbf{x}_T) - \log p(\mathbf{x}_T)] - E_{p(\mathbf{x}_{0:T})}[\log \frac{p(\mathbf{x}_{0:T-1}|\mathbf{x}_T)}{r(\mathbf{x}_{0:T-1}|\mathbf{x}_T)}] \\ &= L_{ELBO} - E_{p(\mathbf{x}_T)}[D_{KL}(p(\mathbf{x}_{0:T-1}|\mathbf{x}_T)||r(\mathbf{x}_{0:T-1}|\mathbf{x}_T))] \\ &\leq L_{ELBO} \end{aligned} \tag{1.6}$$

The performance of this method highly depends on how well the reverse model approximates the reverse dynamics and in practice the reverse model can be specified in some flexible parametric form, like a flexible Neural Network, to achieve good results. However, unless the reverse model approximates the reverse dynamics perfectly, L_{aux} is guaranteed to be a biased approximate of L_{ELBO}. Since the choices of model to approximate the reverse dynamics is limited in practice, L_{aux} may be a loose lower bound of L_{ELBO} and it can significantly affect the performance. Another limitation is that it is expensive to use HMC with multiple iterations in this method because we have to explicitly represent the acceptance decision in Metropolis-Hasting correction step as an additional auxiliary binary random variable ([10]). In particular, HMC used in [9] only involves one iteration with 16 leapfrog steps. Moreover, as the length of the HMC grows, the dimensionality of the auxiliary random variables $\mathbf{x}_{0:T-1}$ increases and $D_{KL}(p(\mathbf{x}_{0:T-1}|\mathbf{x}_T)||r(\mathbf{x}_{0:T-1}|\mathbf{x}_T))$ tends to grow, which makes L_{aux} become looser and looser ([11]). This again prevents us from using longer chain.

Instead of trying to do variational inference with the final states of MCMC, [10] proposed to simply run some HMC steps with initial distribution parameters tuned by standard variational inference. However, they did not propose principled way to tune the HMC hyperparameters. The tuning of these hyperparameters is important for HMC to achieve good performance. Although the distribution of final states of HMC is guaranteed to have lower KL-divergence to the target distribution than the initial distribution does ([12]), we can not be confident that HMC with hyperparameters tuned manually would perform well in complex models ([11]). [13] proposed to use Metropolis-adjusted Langevin dynamics to fit Deep Latent Variable Model (DLVM) in a stochastic Expectation Maximization (EM) fashion. However, like [10], they did not make attempt to tune MCMC hyperparameters. Furthermore, this method requires simulating a Markov Chain for each observation in the dataset, which makes it hard to scale to large datasets ([10]).

More recent works include [14] which is similar to [10] but it attempted to find a better initial distribution than standard variational distribution by defining a new objective L_{VCD} with respect to the initial distribution parameters, which also incorporates feedback from the final states of the Markov Chain:

$$L_{VCD} = D_{KL}(p_0(\mathbf{x})||\pi(\mathbf{x})) - D_{KL}(p_T(\mathbf{x})||\pi(\mathbf{x})) + D_{KL}(p_T(\mathbf{x})||p_0(\mathbf{x})) \tag{1.7}$$

L_{VCD} is a valid divergence and the last term in L_{VCD} cancels the intractable entropy of p_T introduced by the second term, making L_{VCD} tractable. The last term can also be viewed as a regularization which reduces the discrepancy between p_T and p_0. Similar to [10], the drawback of this method is that it also does not tune the MCMC hyperparameters. Moreover, after writing L_{VCD} in an alternative form: $L_{VCD} = E_{p_0(\mathbf{x})}[\log p_0(\mathbf{x}) - \log \pi^*(\mathbf{x})] + E_{p_T(\mathbf{x})}[\log \pi^*(\mathbf{x}) - \log p_0(\mathbf{x})]$, we can see that the second term in this form may slow down the mixing of the Markov Chain since minimizing it encourages the final distribution to be more similar to the initial distribution instead of the target distribution.

The idea of HEI is similar to [10] but it offers a way to tune Markov Chain hyperparameters by simply ignoring the intractable entropy term in L_{ELBO}. Although it may result in unstable performance, we can still often get good results if the initial distribution has sufficient entropy and in this work, we offer techniques to ensure sufficient entropy for initial distribution and thereby improve the performance of HEI.

Chapter 2

Methodology

In this chapter, we propose two techniques to find the initial distribution with sufficient entropy for HEI. The first one is to tune the initial distribution parameters by minimizing α-divergence [8] with a proper divergence parameter α, instead of maximizing L_{ELBO}. The other one is to multiply the marginal variances of the initial distribution found by Variational Inference or α-divergence minimization by a inflation parameter s tuned by optimizing Kernelized Stein Discrepancy (KSD) [15] or a modified version of KSD: Max Sliced Kernelized Stein Discrepancy (maxSKSD) [16].

2.1 α-divergence Minimization

α-divergence is a generalization of KL-divergence and its divergence parameter α controls the width of the approximate distribution $p(\mathbf{x})$. With large positive α, $p(\mathbf{x})$ tends to cover all modes of the target $\pi(\mathbf{x})$ while with small α, $p(\mathbf{x})$ is attracted to the mode of $\pi(\mathbf{x})$ with largest probability mass. There are also two special cases: when $\alpha \to 0$, minimizing α-divergence is equivalent to maximizing L_{ELBO} (VI) while when $\alpha = 1$, minimizing α-divergence is equivalent to Expectation Propagation (EP) [17] (ie. minimize $D_{KL}(\pi(\mathbf{x})||p(\mathbf{x}))$). To demonstrate the differences among the widths of approximate distributions with different divergence parameters α we reproduce the toy example of using a 2D Gaussian with diagonal covariance matrix to approximate the posterior distribution of the weights in a simple 2D linear regression model from [8]. Furthermore, we set the output values of the artificial dataset in a way such that we can compute the solution analytically. The analytical mean of the approximate distribution is $\mathbf{0}$ and the two diagonal elements in the analytical diagonal covariance matrix are the same due to the artificial choice of the output values. We plot the values of the diagonal elements in the covariance matrix, found by analytical computation or stochastic minimization of Monte Carlo estimate of α-divergence, against α in Figure 2.1. We can see from Figure 2.1 that the width of the approximate distribution grows as α increases and this property may be useful to avoid

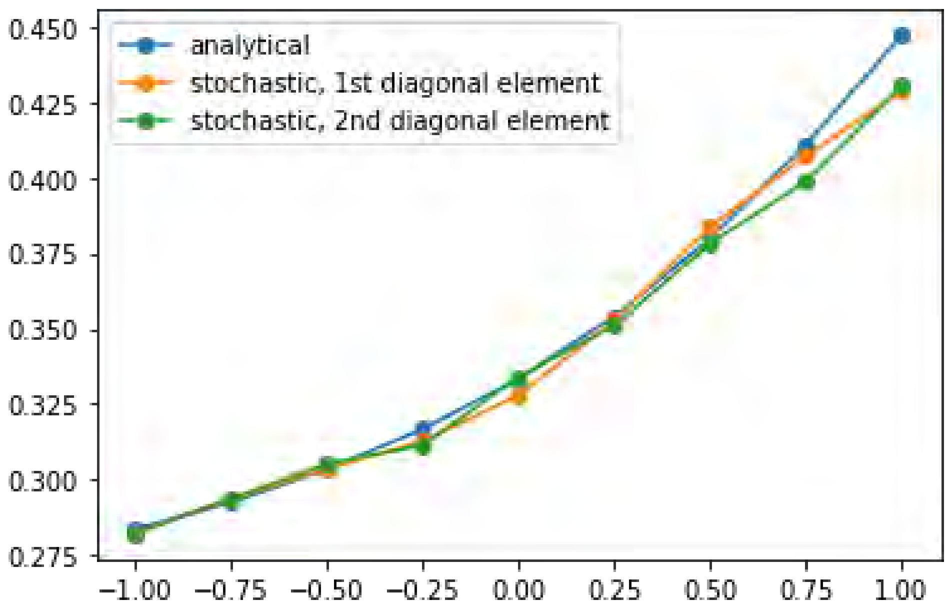


Figure 2.1: Values of diagonal elements of covariance matrix vs α

the mode collapse pathology of HEI since we may find an initial distribution with lager width using α-divergence minimization with a proper α. In particular, with $\alpha > 0$, we can find an initial distribution with larger width than standard variational distribution found by L_{ELBO} maximization.

For our purpose, we minimize α-divergence with $\alpha = 1$ to find a variational initial distribution with larger width than that obtained by L_{ELBO} maximization. It is equivalent to maximizing the following objective: $\{\log E_{P_0(\mathbf{x})}[(\frac{\pi(\mathbf{x})}{P_0(\mathbf{x})})^\alpha]\}|_{\alpha=1} = \log E_{P_0(\mathbf{x})}[\frac{\pi(\mathbf{x})}{P_0(\mathbf{x})}]$. Note that to maximize this objective, we also only need to know the unnormalized target distribution $\pi^*(\mathbf{x})$:

$$L_{\alpha=1} = \log E_{P_0(\mathbf{x})}[(\frac{\pi^*(\mathbf{x})}{P_0(\mathbf{x})})] = \log E_{P_0(\mathbf{x})}[(\frac{\mathcal{Z}\pi(\mathbf{x})}{P_0(\mathbf{x})})] = \log \mathcal{Z} + \log E_{P_0(\mathbf{x})}[(\frac{\pi(\mathbf{x})}{P_0(\mathbf{x})})] \qquad (2.1)$$

Since $\log \mathcal{Z}$ is irrelevant to parameters of P_0, what we really maximize in practice is $L_{\alpha=1}$.

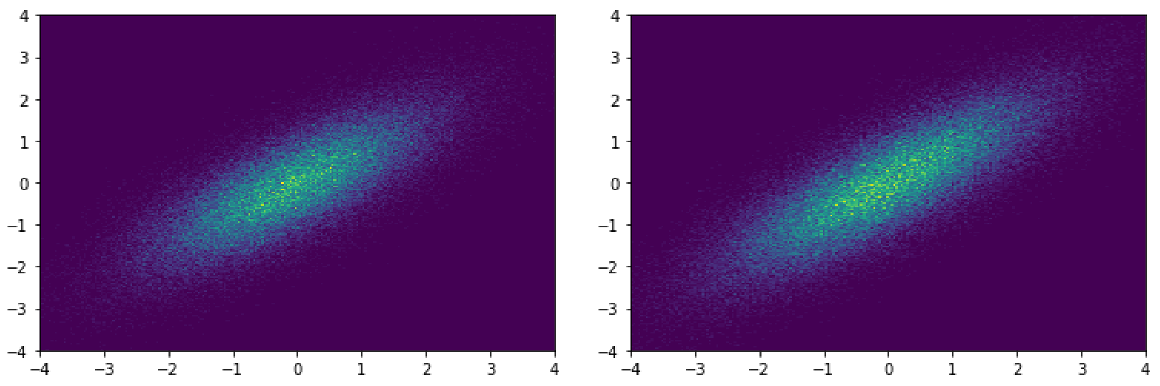
However, direct stochastic minimization of L_α with large α (like $\alpha = 1$) and large number of samples can lead to unstable performance due to poor signal to noise ratio of Monte Carlo gradient estimate of initial distribution parameters ([18]). This can be seen from Figure 2.1: the difference between the estimated values of the diagonal elements and the analytical solution becomes larger as α grows, suggesting the optimization is more unstable as α increases. Therefore, for our experiments, we also construct an alternative version of the gradient estimator with less variance by applying the double reparameterization trick from [19] in case the standard gradient estimator is unstable. Let's use ϕ to denote the initial distribution parameters. $L_{\alpha=1}$ in practice can be written as $E_{\mathbf{x}_{1:K}}[\log(\frac{1}{K}\sum_{i=1}^{K}(\frac{\pi^*(\mathbf{x_i})}{p_\phi(\mathbf{x_i})}))]$, where $\mathbf{x}_{1:K} \sim \prod_i p_\phi(\mathbf{x}_i)$ and $\mathbf{x}_i$ in practice is sampled using reparameterization trick ([20]) to take advantage of the auto-differentiation technique: $\mathbf{x}_i = f(\epsilon_\mathbf{i}, \phi)$, where f is some deterministic function and ϵ is standard Gaussian noise with same dimensionality as $\mathbf{x_i}$. Let's use w_i to denote $\frac{\pi^*(\mathbf{x}_i)}{p_\phi(\mathbf{x}_i)}$. Then $E_{\mathbf{x}_{1:K}}[\log(\frac{1}{K}\sum_{i=1}^{K}(\frac{\pi^*(\mathbf{x}_i)}{p_\phi(\mathbf{x}_i)}))] = E_{\epsilon_{1:K}}[\log(\frac{1}{K}\sum_{i=1}^{K} w_i)]$ and its gradient with respect to ϕ is:

$$\begin{aligned}\nabla_\phi E_{\epsilon_{1:K}}[\log(\frac{1}{K}\sum_{i=1}^{K} w_i)] &= E_{\epsilon_{1:K}}[\sum_1^K(\frac{1}{\sum_{j=1}^K w_j}\nabla_\phi w_i)] \\ &= E_{\epsilon_{1:K}}[\sum_1^K(\frac{w_i}{\sum_{j=1}^K w_j}(-\nabla_\phi \log p_\phi(\mathbf{x}_i) + (\nabla_{\mathbf{x}_i}\log w_i)(\nabla_\phi \mathbf{x}_i)]\end{aligned} \tag{2.2}$$

Note that the gradient contains term $-\nabla_\phi \log p_\phi(\mathbf{x}_i)$, which can contribute significant variance to the gradient estimator ([21]). Thus, we adopt the technique from [19] which treats $\frac{w_i}{\sum_{j=1}^K w_j}(-\nabla_\phi \log p_\phi(\mathbf{x_i}))$ as a REINFORCE gradient term ([22]) and uses an additional application of reparameterization trick to rewrite this term into a more stable form that typically has lower variance (detailed derivation is included in Appendix A):

$$\nabla_\phi E_{\epsilon_{1:k}}[\log(\frac{1}{K}\sum_{i=1}^{K} w_i)] = E_{\epsilon_{1:K}}[\sum_1^K(\frac{w_i}{\sum_{j=1}^K w_j})^2(\nabla_{\mathbf{x}_i}\log w_i)(\nabla_\phi \mathbf{x}_i)] \tag{2.3}$$

We again consider using HEI to generate samples from the correlated bi-variate Gaussian distribution mentioned in section 1.3 ($\pi(\mathbf{x}) \sim \mathcal{N}(\mathbf{0}, [[2.0, 1.5], [1.5, 1.6]])$), but this time we use initial distribution obtained from optimizing $L_{\alpha=1}$ and we consider both cases of using standard gradient estimator (non-DReG) and using Doubly-Reparameterized gradient estimator (DReG) (we call the corresponding objective DReG-$L_{\alpha=1}$). The results are plotted in Figure 2.2. Figure 2.2 shows that with initial distribution tuned by α-divergence minimization with $\alpha = 1$, the pathology is overcome to some extent for this 2D Gaussian example but HEI still does not converge to the target exactly. Furthermore, HEI using DReG version of gradient estimator of initial distribution parameters demonstrates better convergence (ground-truth: $-E_\pi[\log \pi(\mathbf{x})] = 2.8083$) than using standard gradient estimator, suggesting DReG gradient estimator has lower variance and results in more stable performance. This difference can also be seen from the mean parameters of the two initial distributions. The mean of initial distribution found with DReG gradient estimator is $[-0.0118, -0.0078]^T$, which is very close to the mean of the target ($\mathbf{0} = [0, 0]^T$). However, the mean of initial distribution found with standard gradient estimator is $[-0.4492, 0.4697]^T$, which is noticeably different from the mean of the target.



non-DReG, $-E_{P_{30}}[\log \pi(\mathbf{x})] = 2.6615$ DReG, $-E_{P_{30}}[\log \pi(\mathbf{x})] = 2.7328$

Figure 2.2: Histograms of samples generated by HEIs with initial distribution tuned by maximizing non-DReG-$L_{\alpha=1}$ or DReG-$L_{\alpha=1}$

2.2 Kernelized Stein Discrepancy

In case the width of the initial distribution found by $L_{\alpha=1}$ maximization is still not large enough for sufficient entropy, we also incorporate an inflation parameter s into our model. The idea is to multiply the marginal variances of the distribution found by L_{ELBO} maximization (equivalent to $L_{\alpha=0}$ maximization) or $L_{\alpha=1}$ maximization by s to construct an initial distribution with sufficient entropy. We use Kernelized Stein Discrepancy (KSD) [15] to tune s so that it has a proper value that avoids the mode collapse pathology and meanwhile also ensures stable performance. KSD, which has closed-form solution, is a special case of Stein Discrepancy computed in a reproducing kernel Hilbert space (RKHS).

2.2.1 Stein Discrepancy

Stein Discrepancy can be viewed as a special case of integral probability metric (IPM) [23] which is defined as follows:

$$d_{\mathcal{H}}(p, \pi) = \sup_{h \in \mathcal{H}} |E_p[h(\mathbf{x})] - E_\pi[h(\mathbf{x})]| \tag{2.4}$$

where $\mathcal{H}$ is a family of real-valued test functions. IPM measures the maximum discrepancy between expectations with respect to approximate distribution p and target distribution π over functions in class $\mathcal{H}$. Suppose $(p_n)_{n\geq 1}$ is a sequence of sample measures, then the convergence of $d_{\mathcal{H}}(p, \pi)$ to zero implies the $(p_n)_{n\geq 1}$ converges weakly to π, if $\mathcal{H}$ is expressive enough. Note that the term $E_\pi[h(\mathbf{x})]$ in (2.4) is intractable in practical models since it requires us to evaluate an integral under the target $\pi(\mathbf{x})$. We also assume there is no trivial way to compute Monte Carlo estimate of this term in practice since if we can easily generate high quality samples from $\pi(\mathbf{x})$, it is unnecessary for us to find $p(\mathbf{x})$ to approximate $\pi(\mathbf{x})$. However, if we can find a function class $\mathcal{H}$ such that for any $h \in \mathcal{H}$,

$E_\pi[h(\mathbf{x})] = 0$, then we can avoid this intractability. [24] proposed to achieve it by applying the Stein operator of distribution π, $\mathcal{A}_\pi$, to set of functions $\mathcal{F}$ such that for any function $h \in \mathcal{H} = \mathcal{A}_\pi \mathcal{F}$, $E_\pi[h(\mathbf{x})] = 0$ and they called $\mathcal{F}$ satisfying this property Stein set. Moreover, they considered vector-valued functions instead of real-valued functions. The (Langevin) Stein operator $\mathcal{A}_\pi$ of distribution π, with smooth density and support $\mathcal{X} \subset R^D$, acting on vector-valued functions ($\mathbf{f} : \mathcal{X} \to R^D$) is defined as

$$\mathcal{A}_\pi \mathbf{f}(\mathbf{x}) = \mathbf{f}(\mathbf{x})^T \mathbf{s}_\pi(\mathbf{x}) + \nabla_{\mathbf{x}}^T \cdot \mathbf{f}(\mathbf{x}) \tag{2.5}$$

where $\mathbf{s}_\pi(\mathbf{x})$ is the score function of π (derivative of logarithm of $\pi(\mathbf{x})$): $\mathbf{s}_\pi(\mathbf{x}) = \nabla_{\mathbf{x}} \log \pi(\mathbf{x}) = \frac{\nabla_{\mathbf{x}} \pi(\mathbf{x})}{\pi(\mathbf{x})}$ and ∇_x^T is the divergence operator. In particular they defined a function family $\mathcal{F}^*$ called Classical Stein set such that $\mathcal{H} = \mathcal{A}_\pi \mathcal{F}^*$ satisfies the desired property: A function $\mathbf{f} : \mathcal{X} \to R^D$ is said to be in the Classical Stein set if $\mathbf{f}(\mathbf{x})^T \mathbf{n}(\mathbf{x}) = 0, \forall \mathbf{x} \in \partial\mathcal{X}$, where $\mathbf{n}(\mathbf{x})$ is the outward unit normal vector to the boundary $\partial\mathcal{X}$. [24] showed that for any $\mathbf{f}$ in the Classical Stein set $\mathcal{F}$:

$$E_\pi[\mathcal{A}_\pi \mathbf{f}(\mathbf{x})] = 0 \tag{2.6}$$

Therefore, by choosing $\mathcal{H}$ to be function families defined by applying the Stein operator to functions belonging to the Classical Stein set, we can avoid computing the intractable integral with respect to π and we only need to deal with $E_p[\mathcal{A}_\pi \mathbf{f}(\mathbf{x})]$. However, the optimization in general still has no closed form solution since the Classical Stein set is still very expressive, which leads to intractability of the supremum. Thus, we need to further restrict our consideration to a Stein set, which allows the optimization to have closed form solution and meanwhile allows $\mathcal{H}$ to remain sufficiently expressive.

2.2.2 Kernelized Stein Discrepancy

[15] proposed to use a Kernel Stein set to allow the optimization to have closed form solution for Stein Discrepancy. In particular, they considered family of vector-valued functions in the unit ball of vector-valued reproducing kernel Hilbert spaces (RKHS) related to smooth positive definite kernel $k(\mathbf{x}, \mathbf{x}')$ which is in the Stein class of p. The Stein class of p is defined as the family of real-valued functions $f : \mathcal{X} \to R$ satisfying $\int_{\mathbf{x} \in \mathcal{X}} \nabla_x (f(\mathbf{x}) p(\mathbf{x})) d\mathbf{x} = 0$. Note that if the kernel $k(\mathbf{x}, \mathbf{x}')$ is in the Stein Class of p, so is any functions in the real-valued RKHS related to kernel $k(\mathbf{x}, \mathbf{x}')$. The Stein class of p can also be defined for vector-valued functions: we define $\mathcal{F}_p$ as family of vector-valued functions $\mathbf{f} : \mathcal{X} \to R^D$ with each component $f_i : \mathcal{X} \to R$ in the Stein class of p. One can show that the vector-valued test functions from unit norm vector-valued RKHS related to $k(\mathbf{x}, \mathbf{x}')$ form a Stein class of p, $\mathcal{F}_p^k$, and with these test functions, (2.6) is satisfied under

mild conditions (ie. $\mathcal{F}_p^k$ is a Stein set) [25]. Furthermore, the optimization has closed form. In conclusion, by choosing $\mathcal{H}$ to be $\mathcal{A}_\pi \mathcal{F}_p^k$, we can get a special case of Stein Discrepancy with closed form solution for optimization, which is KSD (in fact the formula for KSD given below is the square of the corresponding Stein Discrepancy), $\mathcal{S}^2(p, \pi)$. Suppose π and p are smooth densities and the kernel $k(\mathbf{x}, \mathbf{x}')$ is in the Stein class of p, then

$$\mathcal{S}^2(p, \pi) = \sup_{f \in \mathcal{F}_p^k} E_p^2[\mathcal{A}_\pi \mathbf{f}(\mathbf{x})] = E_{\mathbf{x}, \mathbf{x}' \sim p}[u_\pi(\mathbf{x}, \mathbf{x}')] \tag{2.7}$$

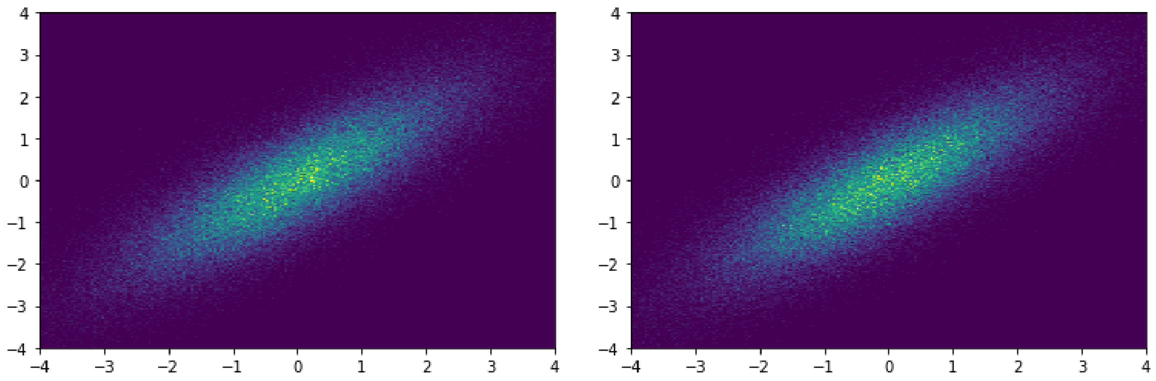
where $u_\pi(\mathbf{x}, \mathbf{x}') = \mathbf{s}_\pi(\mathbf{x})^T k(\mathbf{x}, \mathbf{x}') \mathbf{s}_\pi(\mathbf{x}') + \mathbf{s}_\pi(\mathbf{x})^T \nabla_{\mathbf{x}'} k(\mathbf{x}, \mathbf{x}') + \nabla_{\mathbf{x}} k(\mathbf{x}, \mathbf{x}')^T \mathbf{s}_\pi(\mathbf{x}') +$ $trace(\nabla_{\mathbf{x}, \mathbf{x}'} k(\mathbf{x}, \mathbf{x}'))$. Note $\mathcal{S}^2(p, \pi)$ depends on the target π only through the score function $\mathbf{s}_\pi(\mathbf{x}) = \nabla_{\mathbf{x}} \log \pi(\mathbf{x})$ which can be computed without knowing the normalization constant, and it is a desired property for our purpose since we usually only have unnormalized target π^* in complex models. One can show that $E_p[\mathcal{A}_\pi \mathbf{f}(\mathbf{x})] = E_p[(\mathbf{s}_\pi(x) - \mathbf{s}_p(x))^T \mathbf{f}(\mathbf{x})]$ [24]. Intuitively, this suggests KSD can be seen as an objective comparing the difference between the score functions of π and p. In this work, KSD is estimated using V-statistic as described in [26] with samples drawn from p: $\hat{\mathcal{S}}_v^2(p, \pi) = \frac{1}{n^2} \sum_{i,j} [u_\pi(\mathbf{x}_i, \mathbf{x}_j)]$. It can be shown that if $k(\mathbf{x}, \mathbf{x}')$ is integrally strictly positive definite and $||p(\mathbf{x})(\mathbf{s}_\pi(\mathbf{x}) - \mathbf{s_p}(\mathbf{x}))||_2^2 \leq \infty$ (which may not hold if $p(\mathbf{x})$ has a heavy tail), $\mathcal{S}^2(p, \pi)$ is a valid discrepancy measure between distributions (ie. $\mathcal{S}(p, \pi) \geq 0$ and $\mathcal{S}^2(p, \pi) = 0$ if and only if $p = \pi$). The kernel $k(\mathbf{x}, \mathbf{x}')$ used in this work is the RBF kernel, $k(\mathbf{x}, \mathbf{x}') = \exp(-\frac{1}{2h^2} ||\mathbf{x} - \mathbf{x}'||_2^2)$, which is in the Stein class of distributions that have smooth densities supported on $\mathcal{R}^D$. The bandwidth of the kernel h is chosen to be the median of the data distances.

Despite of the nice theoretical results described above, the expressive power of RKHS declines when dimension of the state space increases ([27]) and more samples may be required to estimate KSD in high dimension, which limits its ability to scale to high dimensional data since the cost of computing KSD is quadratic in sample size, which becomes impractical quickly as the sample size increases. Furthermore, KSD based on common kernels such as RBF kernel and Matern kernel may not be able to detect non-convergence in high dimensions ([25]). We will describe in next section a modified version of KSD proposed in [16], which can mitigate some of the curse-of-dimensionality problems of KSD.

The training scheme of HEI-KSD can now be summarized as follows:

$$\begin{aligned} &\max_\psi \text{DReG-}L_{\alpha=1} \text{ or } \max_\psi L_{ELBO} \text{ (equivalent to } \max_\psi L_{\alpha=0}) \\ &\max_\theta L_{EI} \\ &\min_s \mathcal{S}^2(P_T, \pi^*) \end{aligned} \tag{2.8}$$

where θ is the HMC (with T iterations) hyperparameters and ψ is the initial distribution parameters. We again use the 2D Gaussian example to carry out a simple test for this new training scheme. We consider initial distributions tuned by maximizing L_{ELBO} (maximizing $L_{\alpha=0}$) or by maximizing DReG-$L_{\alpha=1}$.



$\alpha = 0$, $-E_{P_{30}}[\log \pi(x)] = 2.7990$, $s = 5.8869$ $\alpha = 1$, $-E_{P_{30}}[\log \pi(x)] = 2.8079$, $s = 1.3762$

Figure 2.3: Histograms of samples generated by HEI-KSDs with initial distributions tuned by maximizing L_{ELBO} (maximizing $L_{\alpha=0}$) or maximizing DReG-$L_{\alpha=1}$

Figure 2.3 shows the results generated by HEI-KSD. We can see that HEI-KSD successfully avoids the pathology and is able to converge to the target distribution (ground-truth: $-E_{\pi}[\log \pi(\mathbf{x})] = 2.8083$) no matter which method is used to tune the initial distribution, suggesting the performance of HEI indeed can be improved by incorporating the inflation parameter tuned by KSD into the model.

2.3 Max Sliced Kernelized Stein Discrepancy

As mentioned before, KSD suffers from curse-of-dimensionality. To address this problem, [16] proposed a modified version of KSD, which is called Max Sliced Kernelized Stein Discrepancy (maxSKSD), and it also has a closed form solution for optimization.

2.3.1 Sliced Stein Discrepancy

To derive maxSKSD, we first introduce Sliced Stein Discrepancy (SSD). There are two sources that contribute to the curse-of-dimensionality problem: one of them is the score function $\mathbf{s}_{\pi}(\mathbf{x})$ and the other one is the test function $\mathbf{f}(\mathbf{x})$, where $\mathbf{x}$ is supported on $\mathcal{X} \subset R^D$. [16] dealt with the first source by projecting $\mathbf{s}_{\pi}(\mathbf{x})$ onto a slicing direction $\mathbf{r}$ ($\mathbf{r} \in R^D$): $s_{\pi}^{\mathbf{r}}(\mathbf{x}) = \mathbf{s}_{\pi}(\mathbf{x})^T\mathbf{r}$, which is equivalent to slicing the target π through the slicing direction at $\mathbf{x}$. Then it can be proved that $\pi = p$ a.e. if and only if $s_{\pi}^{\mathbf{r}}(\mathbf{x}) = s_{p}^{\mathbf{r}}(\mathbf{x})$ for all $\mathbf{r}$. The second source is addressed by projecting the input of test function $\mathbf{f}$, $\mathbf{x}$, to reduce the dimensionality of the input. Note that this time we can not project $\mathbf{x}$ onto $\mathbf{r}$ again since this will cost information loss as the optimal test function corresponding to the projected score is proportional to $s_\pi^{\mathbf{r}}(\mathbf{x}) - s_p^{\mathbf{r}}(\mathbf{x})$. Instead, we project $\mathbf{x}$ onto infinitely many test directions $\mathbf{g}$ and then take the average over $\mathbf{g}$. More specifically, suppose π and p are two D-dimensional distributions and we define the test functions $f(\mathbf{x};\mathbf{r},\mathbf{g}) : R^D \to R$ to be $f_{\mathbf{rg}}(\mathbf{x}^T\mathbf{g})$, where $f_{\mathbf{rg}}$ takes one-dimensional input, then the SSD between p and π is

$$S(p,\pi) = E_{p_{\mathbf{r}},p_{\mathbf{g}}}[\sup_{f_{\mathbf{rg}}\in\mathcal{F}_p} E_p[s_\pi^{\mathbf{r}}(\mathbf{x})f_{\mathbf{rg}}(\mathbf{x}^T\mathbf{g}) + \mathbf{r}^T\mathbf{g}\nabla_{\mathbf{x}^T\mathbf{g}}f_{\mathbf{rg}}(\mathbf{x}^T\mathbf{g})]] \tag{2.9}$$

where $p_{\mathbf{r}}(\mathbf{r})$ and $p_{\mathbf{g}}(\mathbf{g})$ are two uniform distributions over the hypersphere $\mathcal{S}^{D-1}$ respectively. $\mathcal{F}_p$ denotes the Stein Class of p and $f_{\mathbf{rg}} \in \mathcal{F}_p$ represents $f(\cdot;\mathbf{r},\mathbf{g}) \in \mathcal{F}_p$. One can show that SSD is a valid discrepancy between two distributions under mild conditions. However, for practical high-dimensional models, the standard Monte-Carlo estimate of $S(p,\pi)$ requires us to average over samples of $\mathbf{r}$ and $\mathbf{g}$ which live in high-dimensional hypersphere and this usually requires a very large sample size for accurate estimation. To address this problem of scalability, [16] showed that one can relax the requirement of considering infinitely many slicing directions by using finitely many slicing directions from an orthogonal basis $O_{\mathbf{r}}$ of R^D, such as the standard orthonormal basis consisting of D one-hot vectors, and the computational cost can be further reduced by only using the optimal test direction $\mathbf{g}_{\mathbf{r}}$ corresponding to each slicing direction $\mathbf{r}$. The resulting relaxed SSD is called maxSSD and its formula is shown as follows:

$$S_{max}(p,\pi) = \sum_{\mathbf{r}\in O_{\mathbf{r}}} \sup_{f_{\mathbf{rg_r}}\in\mathcal{F},\mathbf{g_r}\in S^{D-1}} E_p[s_\pi^{\mathbf{r}}(\mathbf{x})f_{\mathbf{rg_r}}(\mathbf{x}^T\mathbf{g_r}) + \mathbf{r}^T\mathbf{g_r}\nabla_{\mathbf{x}^T\mathbf{g_r}}f_{\mathbf{rg_r}}(\mathbf{x}^T\mathbf{g_r})] \tag{2.10}$$

The maxSSD can also be shown as a valid discrepancy under mild conditions. As discussed previously, using the slicing direction as the test direction can cause trouble and it is necessary for us to find optimal $\mathbf{g}$ for each $\mathbf{r}$, $\mathbf{g_r}$, when computing maxSSD. This can be demonstrated by the counterexample in [16], in which $O_{\mathbf{r}}$ is the standard orthonormal basis of R^D and $\mathbf{g}$ is set to be equal to $\mathbf{r}$ for each $\mathbf{r}$. It is found that the corresponding discrepancy under this setting between two distinct distributions with same marginals is not able to detect the difference between the two distributions, and thus the discrepancy in this case is not valid.

2.3.2 Sliced Kernelized Stein Discreancy

SSD is still infeasible due to the intractability of supremum without further restriction on the family of test functions. Thus, we apply the kernel trick again to obtain a closed form solution. In detail, suppose $\mathcal{H}_{\mathbf{rg}}$ is a real-valued RKHS related to kernel $k(\mathbf{x},\mathbf{x}';\mathbf{r},\mathbf{g}) = k_{\mathbf{rg}}(\mathbf{x}^T\mathbf{g},\mathbf{x}'^T\mathbf{g})$, which is in the Stein Class of p and bounded for all $\mathbf{r}$ and

$\mathbf{g}$, then we introduce following quantities:

$$\begin{aligned}\xi_{\pi,\mathbf{r},\mathbf{g}}(\mathbf{x},\cdot) &= s_\pi^{\mathbf{r}}(\mathbf{x})k_{\mathbf{rg}}(\mathbf{x}^T\mathbf{g},\cdot) + \mathbf{r}^T\mathbf{g}\nabla_{\mathbf{x}^T\mathbf{g}}k_{\mathbf{rg}}(\mathbf{x}^T\mathbf{g},\cdot)\\ h_{\pi,\mathbf{r},\mathbf{g}}(\mathbf{x},\mathbf{y}) &= s_\pi^{\mathbf{r}}(\mathbf{x})k_{\mathbf{rg}}(\mathbf{x}^T\mathbf{g},\mathbf{y}^T\mathbf{g})s_\pi^{\mathbf{r}}(\mathbf{y}) + \mathbf{r}^T\mathbf{g}s_\pi^{\mathbf{r}}(\mathbf{y})\nabla_{\mathbf{x}^T\mathbf{g}}k_{\mathbf{rg}}(\mathbf{x}^T\mathbf{g},\mathbf{y}^T\mathbf{g}) + \\ &\quad \mathbf{r}^T\mathbf{g}s_\pi^{\mathbf{r}}(\mathbf{x})\nabla_{\mathbf{y}^T\mathbf{g}}k_{\mathbf{rg}}(\mathbf{x}^T\mathbf{g},\mathbf{y}^T\mathbf{g}) + (\mathbf{r}^T\mathbf{g})^2\nabla^2_{\mathbf{x}^T\mathbf{g},\mathbf{y}^T\mathbf{g}}k_{\mathbf{rg}}(\mathbf{x}^T\mathbf{g},\mathbf{y}^T\mathbf{g})\end{aligned} \tag{2.11}$$

With standard orthonormal basis of R^D and given $E_p[h_{\pi,\mathbf{r},\mathbf{g}}(\mathbf{x},\mathbf{x})] < \infty$ and some other mild conditions, the Sliced Kernelized Stein Discrepancy (SKSD) between two D-dimensional distributions p and π is then defined as:

$$SK_o(p,\pi) = \sum_{\mathbf{r}\in O_{\mathbf{r}}} \int_{S^{D-1}} p_{\mathbf{g}}(\mathbf{g}) D^2_{\mathbf{rg}}(p,\pi) d\mathbf{g} \tag{2.12}$$

where $D^2_{\mathbf{rg}}(p,\pi)$ has closed form solution: $D^2_{\mathbf{rg}}(p,\pi) = ||\sup_{f_{\mathbf{rg}}\in\mathcal{H}_{\mathbf{rg}},||f_{\mathbf{rg}}||\le 1} E_p[s_\pi^{\mathbf{r}}(\mathbf{x})f_{\mathbf{rg}}(\mathbf{x}^T\mathbf{g}) + \mathbf{r}^T\mathbf{g}\nabla_{\mathbf{x}^T\mathbf{g}}f_{\mathbf{rg}}(\mathbf{x}^T\mathbf{g})]||^2 = ||E_p[\xi_{\pi,\mathbf{r},\mathbf{g}}(\mathbf{x})]||^2_{\mathcal{H}_{\mathbf{rg}}} = E_{\mathbf{x},\mathbf{x}'\sim p}[h_{\pi,\mathbf{r},\mathbf{g}}(\mathbf{x},\mathbf{x}')]$. $SK_o(p,\pi)$ is a valid discrepancy under mild conditions. Like SSD, we can only use the optimal test direction $\mathbf{g_r}$ corresponding to each slicing direction $\mathbf{r}$ instead of computing integral over $\mathbf{g}$ to reduce the computational cost. The resulting discrepancy is named maxSKSD and its equation is given below:

$$SK_{max}(p,\pi) = \sum_{\mathbf{r}\in O_{\mathbf{r}}} \sup_{\mathbf{g_r}} D^2_{\mathbf{rg_r}}(p,\pi) \tag{2.13}$$

For maxSKSD, the kernel used is again the RBF kernel with bandwidth chosen according to median heuristic and we estimate maxSKSD again using V-statistic: $\hat{SK}_{max,v}(p,\pi) = \frac{1}{N^2}\sum_{\mathbf{r}\in O_{\mathbf{r}}}\sum_{i,j} h_{\pi,\mathbf{r},\mathbf{g_r}}(\mathbf{x_i},\mathbf{x_j})$, where $\{\mathbf{x_i}\}_1^N$ is a set of i.i.d samples drawn from p.

Now we have HEI-maxSKSD model which tunes the inflation parameter s by minimizing maxSKSD and we carry out a simple test for this model by using it to generate samples again from the 2D Gaussian density described in previous sections. Figure 2.4 shows the histograms of samples generated by HEI-maxSKSD and one can see that the results are close to the ground-truth ($-E_\pi[\log\pi(\mathbf{x})] = 2.8083$) no matter which objective (L_{ELBO} ($\alpha = 0$) or DReG-$L_{\alpha=1}$) is used for tuning the initial distribution parameters. It is also worth noting that each test direction is very close to its corresponding slicing direction (for $\alpha = 0$, $\mathbf{g1} = [0.9993, 0.0375]^T, \mathbf{g2} = [-0.1609, 0.9870]^T$ and for $\alpha = 1$, $\mathbf{g1} = [0.9961, 0.0882]^T, \mathbf{g2} = [0.0219, 0.9998]^T$), suggesting that for this example, finding the optimal inflation parameter using maxSKSD may be seen as matching the marginals between P_{30} and π.

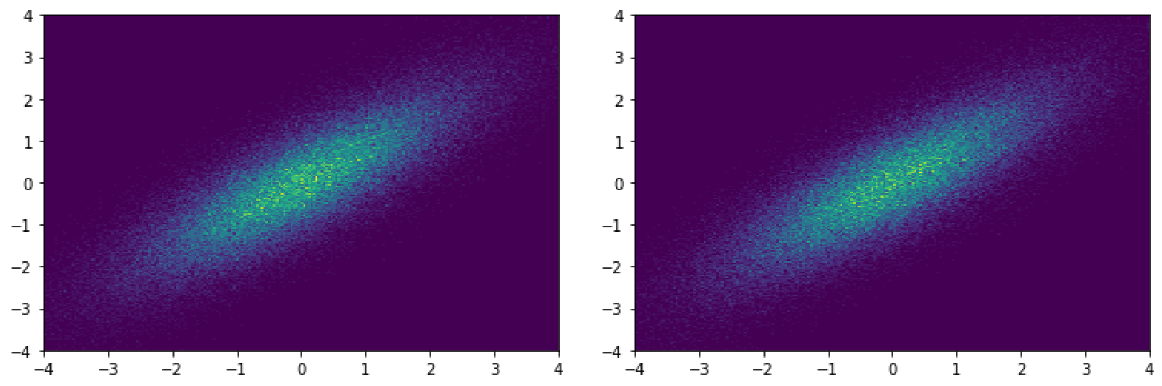


Figure 2.4: Histograms of samples generated by HEI-maxSKSDs with initial distributions tuned by maximizing L_{ELBO} or maximizing DReG-$L_{\alpha=1}$

2.4 Related Work

KSD was first proposed to develop a Goodness of Fit test ([15, 26]) and in recent years, many researches explore opportunities of applying KSD to solve other inferential tasks and develop corresponding techniques. Techniques related to KSD are applied in various area, such as Variational Inference ([28]) and training generative adversarial networks ([29]). In this section, we discuss two techniques related to KSD: Kernel Test of Goodness of Fit and Stochastic Variational Gradient Descent. Note that these techniques can also be developed based on maxSKSD as demonstrated in [16].

2.4.1 Kernel Test of Goodness of Fit

KSD can be used to construct a Goodness of Fit test (GOF) with null hypothesis: $H_0 : p = \pi$, which can be used to assess how well the approximate distribution p in the model matches the true target π. In particular, [15] used U-statistic of KSD, $\hat{S}_u(p, \pi)$, to achieve this task:

$$\hat{S}_u(p, \pi) = \frac{1}{n(n-1)} \sum_{1 \leq i \neq j \leq n} u_\pi(\mathbf{x_i}, \mathbf{x_j}) \tag{2.14}$$

where $u_\pi(\mathbf{x}, \mathbf{x}') = s_\pi(\mathbf{x})^T k(\mathbf{x}, \mathbf{x}') s_\pi(\mathbf{x}') + s_\pi(\mathbf{x})^T \nabla_{\mathbf{x}'} k(\mathbf{x}, \mathbf{x}') + \nabla_{\mathbf{x}} k(\mathbf{x}, \mathbf{x}')^T s_\pi(\mathbf{x}') + trace(\nabla_{\mathbf{x},\mathbf{x}'} k(\mathbf{x}, \mathbf{x}'))$. Note that U-statistic is an unbiased estimator of KSD but it may be negative, while V-statistic is always non-negative but it is biased. [15] showed that $\hat{S}_u(p, \pi)$ has well-defined limit distribution under mild conditions:

1. If $p \neq \pi$, then $\sqrt{n}(\hat{S}_u(p, \pi) - S(p, \pi)) \xrightarrow{d} \mathcal{N}(0, \sigma_u^2)$, where $\sigma_u^2 = Var_{\mathbf{x} \sim p}(E_{\mathbf{x}' \sim p}[u_\pi(\mathbf{x}, \mathbf{x}')]) \neq 0$
2. If $p = \pi$, then we have a degenerated U-statistic with $\sigma_u = 0$ and $n\hat{S}_u(p, \pi) \xrightarrow{d} \sum_{j=1}^{\infty} c_j (Z_j^2 - 1)$, where $\{Z_j\}$ are i.i.d standard Gaussian random variables and $\{c_j\}$ are

eigenvalues of kernel $u_\pi(\mathbf{x}, \mathbf{x}')$ under p.

Based on the above theoretical result, we can construct a bootstrap GOF test based on KSD. We first compute $\hat{S}_u(p, \pi)$ according to (2.14). Then we draw random weights from multinomial distributions $\{(w_1^m, ..., w_n^m)\}_{m=1}^M \sim Mult(n; \frac{1}{n}, ..., \frac{1}{n})$ and compute the bootstrap samples $\hat{S}_u^m(p, \pi) = \sum_{i \neq j}(w_i^m - \frac{1}{n})(w_j^m - \frac{1}{n})u_\pi(\mathbf{x_i}, \mathbf{x_j})$, for $m = 1, ..., M$. With these quantities, we can reject the null hypothesis $H_0 : p = \pi$ if the proportion of the bootstrap samples $\hat{S}_u^m(p, \pi)$ that are greater than $\hat{S}_u(p, \pi)$ is less than the significance level α.

2.4.2 Stein Variational Gradient Descent

Instead of direct minimization of KSD, [30] proposed a particle inference algorithm based on KSD, called Stein Variational Gradient Descent (SVGD), which relates KSD to the gradient of KL divergence between approximate distribution p and the target π. SVGD defines a series of deterministic mapping for a set of particles, which leads to steepest descent of KL divergence in RKHS $\mathcal{H}_d$ from particles' underlying distribution p to the target π.

Given a perturbation $\phi(\mathbf{x})$, let $T(\mathbf{x}) = \mathbf{x} + \epsilon\phi(\mathbf{x})$, where $\mathbf{x} \sim p(\mathbf{x})$, and the density of $\mathbf{z} = T(\mathbf{x})$ be $p_{[T]}(\mathbf{z})$, then we have

$$\nabla_\epsilon D_{KL}(p_{[T]}||\pi)|_{\epsilon=0} = -E_p[\mathcal{A}_\pi \phi(\mathbf{x})] \tag{2.15}$$

From the above equation, we can see that if the perturbation is in RKHS, then the optimal test function in KSD is the optimal perturbation function. In detail, the steepest descent direction in KL divergence is given by

$$\phi^*(\cdot) = E_p[\nabla_\mathbf{x} \log \pi(\mathbf{x}) k(\mathbf{x}, \cdot) + \nabla_\mathbf{x} k(\mathbf{x}, \cdot)] \tag{2.16}$$

Note that $\pi(\mathbf{x})$ in (2.16) can be replaced by the unnormalized target $\pi^*(\mathbf{x})$ since $\nabla_\mathbf{x} \log \pi(\mathbf{x}) = \nabla_\mathbf{x} \log \pi^*(\mathbf{x})$. In practice, (2.16) can be estimated by Monte Carlo, starting from a set of initial particles. The first term in (2.16) with $\nabla_\mathbf{x} \log \pi(\mathbf{x})$ drives the particle towards modes of the target while the second term encourages diversity. When descent in KL divergence equals to zero, the particles stop moving, and KSD equals to zero, implying $p(\mathbf{x}) = \pi(\mathbf{x})$ a.e.. It is worth noting that in the special case where there is only one particle and $\nabla_\mathbf{x} k(\mathbf{x}, \mathbf{x}') = 0$ when $\mathbf{x} = \mathbf{x}'$, SVGD is equivalent to maximizing $\log \pi(\mathbf{x})$ with gradient ascent.

Chapter 3

2D Experiments Analysis

In this chapter, we demonstrate the application of HEI-KSD and HEI-maxSKSD to generate samples from various bivariate distributions. In particular, we consider six benchmark 2D densities (the log-unnormalized-probability density functions of the six distributions are shown in Appendix B) and their ground-truths are estimated by Rejection Sampling. Figure 3.1 shows the histograms of these six distributions and Table 3.1 shows their corresponding $-E_\pi[\log \pi^*(\mathbf{x})]$.

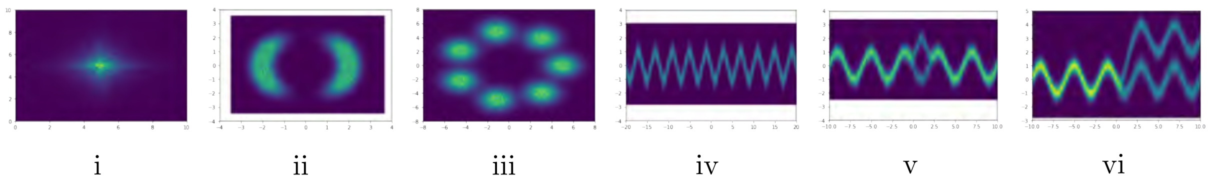


Figure 3.1: Ground-truth histograms of six bivariate distributions generated by rejection sampling

	i	ii	iii	iv	v	vi
$-E_\pi[\log \pi^*(\mathbf{x})]$	2.0075	0.8511	0.9282	0.4994	-0.1703	0.1483

Table 3.1: Ground truth $-E_\pi[\log \pi^*(\mathbf{x})]$ estimated by rejection sampling for six bivariate distributions

3.1 Setup of Experiments

The initial distributions for HMC are assumed to be factorized Gaussian and we consider finding the initial distributions in two ways by either maximizing L_{ELBO} ($\alpha = 0$) or maximizing DReG-$L_{\alpha=1}$. The HMC used contains 30 iterations and each iteration includes 5 leapfrog steps. We consider different step sizes and momentum variances for each HMC iteration and for each dimension of samples. The step sizes and momentum variances are tuned by maximizing $L_{EI} = E_{P_{30}}[\pi^*(\mathbf{x})]$. We use Adam [31], with hyperparameters $\beta_1 = 0.9$, $\beta_2 = 0.99$, $\epsilon = 10^{-8}$, as the stochastic optimization algorithm. For each distribution, we run Adam either for 500 parameter updates with step size 0.02 or for 200 parameter updates with step size 0.05. However, for each distribution, we fix the number of parameter updates and Adam step size across all different training schemes. The HMC step sizes are randomly initialized according to a uniform distribution over $(0.01, 0.025)$ and the momentum variances are all initialized to be 1. The mean and covariance of initial distribution P_0 are initialized to be $\mathbf{0}$ and $\mathbf{I}$ respectively. To evaluate the performance of different models in 2D experiments, we compare the histograms generated by different models with the ground-truth histograms and we also compare $-E_{P_{30}}[\log \pi^*(\mathbf{x})]$ with ground-truth $-E_{\pi}[\log \pi^*(\mathbf{x})]$ estimated by rejection sampling. We use 100000 samples generated from each model to plot the corresponding histogram and estimate $-E_{P_{30}}[\log \pi^*(\mathbf{x})]$.

3.2 Experimental Results

We consider six different models: (a) vanilla HEI with initial distribution tuned by maximizing L_{ELBO}, (b) HEI with initial distribution tuned by maximizing DReG-$L_{\alpha=1}$, (c) HEI-KSD with initial distribution tuned by maximizing L_{ELBO}, (d) HEI-maxSKSD with initial distribution tuned by maximizing L_{ELBO}, (e) HEI-KSD with initial distribution tuned by maximizing DReG-$L_{\alpha=1}$, (f) HEI-maxSKSD with initial distribution tuned by maximizing DReG-$L_{\alpha=1}$.

	i	ii	iii	iv	v	vi
(a)	1.9944	1.0315	0.9183	0.4846	0.0435	0.1345
(b)	2.0431	2.6602	0.9197	0.4981	-0.0510	0.6757
(c)	2.0036	0.8044	0.9191	0.4933	-0.1897	0.1640
(d)	2.0034	0.8719	0.9183	0.4983	-0.1813	0.2614
(e)	2.0834	1.4890	0.9205	0.4999	-0.1284	0.1547
(f)	2.0171	1.4476	0.9190	0.4986	-0.1375	0.6036

Table 3.2: $-E_{P_{30}}[\log \pi^*(\mathbf{x})]$ of different models for six bivariate distributions

	i	ii	iii	iv	v	vi
(c)	1.8141	0.4441	4.8171	7.6005	22.8417	0.4812
(d)	1.4636	0.6521	3.1584	20.4322	12.0305	4.6885
(e)	2.9839	0.9014	0.1357	7.3995	4.2350	1.0694
(f)	2.4196	0.8354	3.6206	4.7707	4.8167	0.7032

Table 3.3: optimal inflation value s of HEI-KSD/maxSKSD models for six bivariate distributions

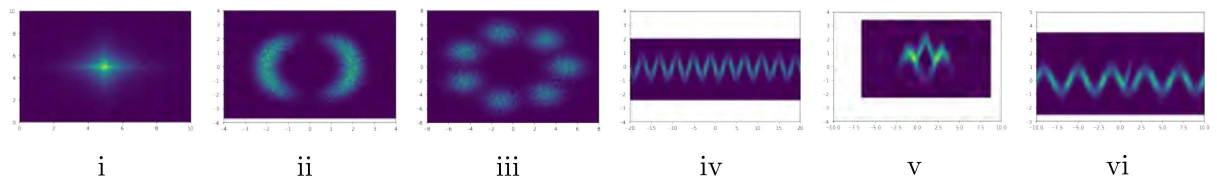


Figure 3.2: Histograms of six bivariate distributions generated by model a (vanilla HEI, $\alpha = 0$)

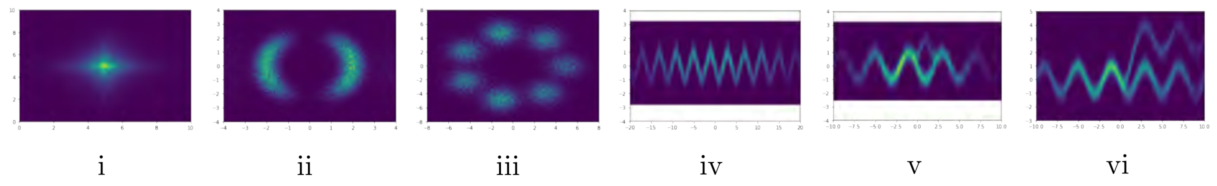


Figure 3.3: Histograms of six bivariate distributions generated by model b (vanilla HEI, $\alpha = 1$)

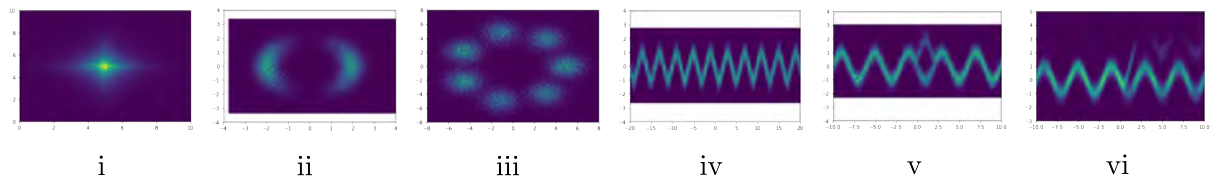


Figure 3.4: Histograms of six bivariate distributions generated by model c (HEI-KSD, $\alpha = 0$)

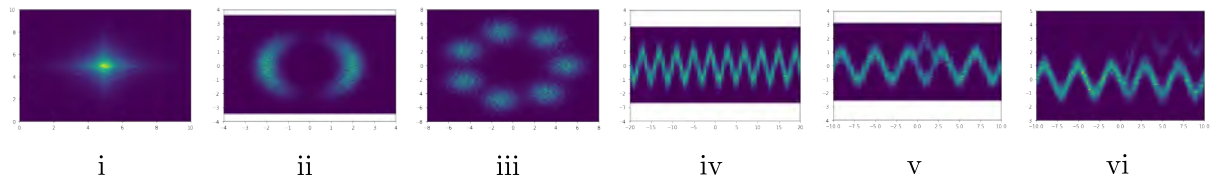


Figure 3.5: Histograms of six bivariate distributions generated by model d (HEI-maxSKSD, $\alpha = 0$)

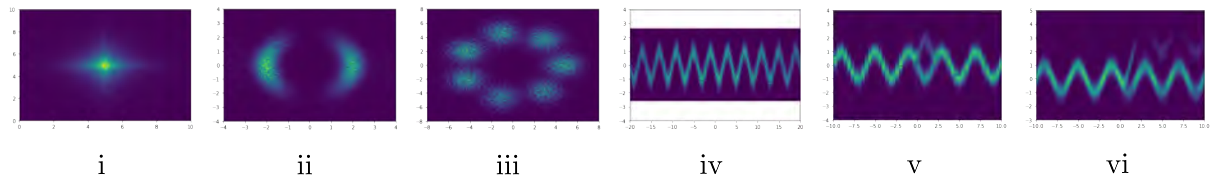


Figure 3.6: Histograms of six bivariate distributions generated by model e (HEI-KSD, $\alpha = 1$)

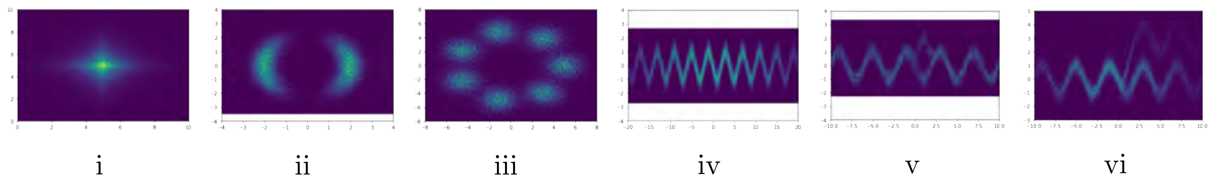


Figure 3.7: Histograms of six bivariate distributions generated by model f (HEI-maxSKSD, $\alpha = 1$)

Table 3.2 shows $-E_{P_{30}}[\log \pi^*(\mathbf{x})]$ estimated by different models and Table 3.3 shows the optimal inflation values applied to the marginal variances of initial distributions for HEI-KSD or HEI-maxSKSD models (c, d, e and f). The histograms of the six distributions generated by model (a), (b), (c), (d), (e) and (f) are plotted in Figure 3.2, 3.3, 3.4, 3.5, 3.6 and 3.7 respectively.

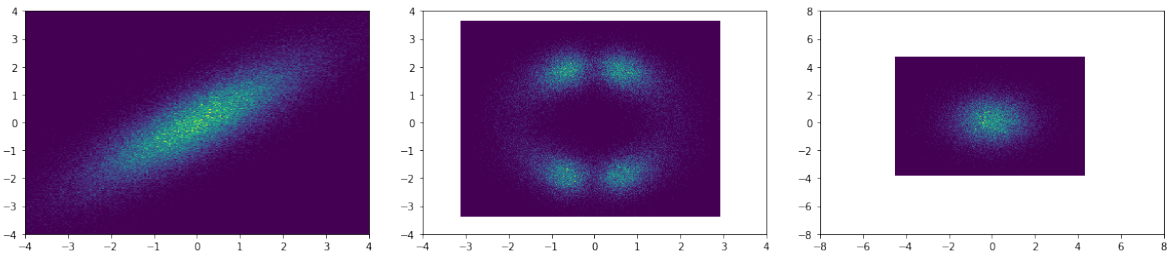
Comparing these results with the ground-truth shown previously, one can see that HEI-KSD/maxSKSD models (c, d, e and f) are capable of preventing the mode collapse pathology (especially for distribution iv and v), and in general they result in better convergence than HEI models without inflation parameter (a, b). For distributions that are relatively easy for model a and b to approximate, such as i, iii and iv, model c, d, e, and f also work very well. However, for distributions that model a and b perform poorly, model c, d, e and f can still achieve good performance. For example, model a and b both fail to converge to target distribution v due to mode collapse pathology, however, model c, d, e and f all avoid the pathology and result in good convergence.

In general, HEI-KSD or HEI-maxSKSD with initial distribution tuned by maximizing $L_{\alpha=0}$ (model c and d) achieve higher performance than the other models and in particular, model c shows good convergence for all six distributions. Although model e and f perform well for most distributions, they both overestimate $-E_{P_{30}}[\log \pi^*(\mathbf{x})]$ for distribution ii and model f overestimates $-E_{P_{30}}[\log \pi^*(\mathbf{x})]$ for distribution vi, suggesting tuning initial distribution with DReG-$L_{\alpha=1}$ maximization might still lead to unstable performance.

HEI-maxSKSD shows no advantage against HEI-KSD in these 2D experiments. Instead, HEI-KSD performs slightly better than HEI-maxSKSD in these 2D experiments. For example, model (c) and (e) show better convergence for distribution vi than model (d) and (f) respectively, suggesting in low dimension problems, KSD can work well and models using maxSKSD may not outperform models using KSD.

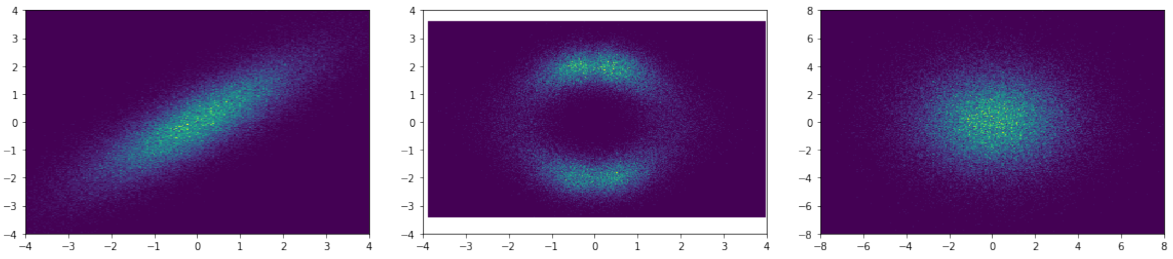
It is also worth noting that the optimal test directions $\mathbf{G} = [\mathbf{g1}, \mathbf{g2}]^T$ of HEI-maxSKSD models are close to the standard orthonormal basis of R^2, suggesting tuning inflation parameter with maxSKSD is similar to matching the marginals between P_{30} and π for these 2D experiments. Moreover, if we fix $\mathbf{G}$ to be the standard orthonormal basis of R^2 during optimization, the results are comparable to that of HEI-maxSKSD. Although the corresponding objective might be useful in some cases, it is not a valid discrepancy measure, as shown in [16].

3.3 Advantage of L_{EI} Against KSD and maxSKSD



$-E_{P_{30}}[\log \pi^*(x)] = 2.8137$ $\quad -E_{P_{30}}[\log \pi^*(x)] = 2.5945$ $\quad -E_{P_{30}}[\log \pi^*(x)] = 6.8780$

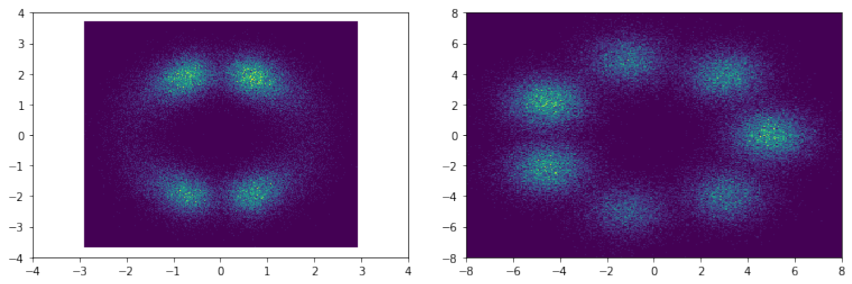
Figure 3.8: Histograms of three bivariate distributions generated by model that tuning HMC hyperparameters and inflation by minimizing KSD (model g). $\alpha = 0$



$-E_{P_{30}}[\log \pi^*(x)] = 2.7877$ $\quad -E_{P_{30}}[\log \pi^*(x)] = 2.8610$ $\quad -E_{P_{30}}[\log \pi^*(x)] = 3.9807$

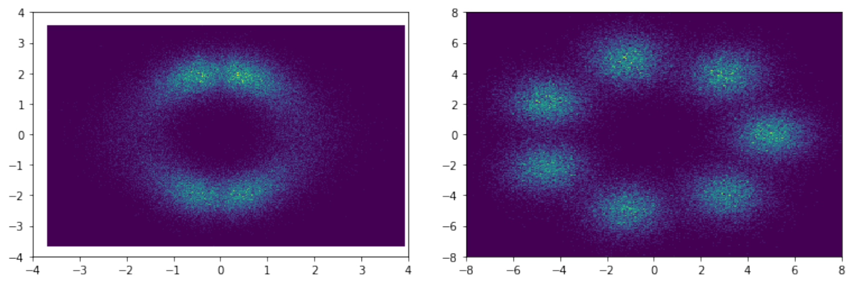
Figure 3.9: Histograms of three bivariate distributions generated by model that tuning HMC hyperparameters and inflation by minimizing maxSKSD (model h). $\alpha = 0$

Instead of tuning HMC hyperparameters with L_{EI} maximization, they can also be tuned by minimizing KSD or maxSKSD along with the inflation parameter. In some cases, it can give us comparable result as that obtained by HEI-KSD/maxSKSD. However, we find that tuning HMC hyperparameters and inflation s with KSD or maxSKSD minimization can result in unstable performance for complex distributions. To demonstrate it, we show results obtained by tuning HMC hyperparameters and inflation parameter with KSD or maxSKSD for the correlated Gaussian described in previous chapters, distribution ii and distribution iii (the initial distributions are tuned by maximizing L_{ELBO}). The ground-truth $-E_{P_{30}}[\log \pi^*(\mathbf{x})]$ for these three distributions are 2.8083, 0.8511 and 0.9282 respectively. We refer model that tuning HMC hyperparameters and inflation parameter with KSD as model g and model that tuning HMC hyperparameters and inflation parameter with maxSKSD as model h.



$-E_{P_{30}}[\log \pi^*(x)] = 2.5174$ $-E_{P_{30}}[\log \pi^*(x)] = 0.9192$

Figure 3.10: Histograms of distribution ii and iii generated by model g with pretrained parameters from model c as initialization. $\alpha = 0$



$-E_{P_{30}}[\log \pi^*(x)] = 2.7317$ $-E_{P_{30}}[\log \pi^*(x)] = 0.9148$

Figure 3.11: Histograms of distribution ii and iii generated by model h with pretrained parameters from model d as initialization. $\alpha = 0$

Figure 3.8 and 3.9 show results of model g and model h for these three distributions respectively. For relatively simple distributions, like correlated Gaussian, model g and model h both can work well. However, both models fail to converge to the target for more complex distributions, like distribution ii and iii, suggesting tuning many parameters with KSD or maxSKSD can lead to poor performance. Furthermore, even with good parameters initialization, model g and h can diverge. For distribution ii and iii, Figure 3.10 and 3.11 show results of model g and model h with pretrained HEI-KSD (model c in previous section) parameters and pretrained HEI-maxSKSD (model d in previous section) parameters as initialization respectively. With good initial parameters, model g and h both are able to converge for distribution iii, but both of them still diverge for distribution ii. It further implies L_{EI} is a better objective to use for tuning HMC hyperparameters. L_{EI} has power of driving samples from the approximate distribution close to the modes of the target. While KSD or maxSKSD lack this power, they instead seek to find approximate distribution with score function similar to that of the target distribution, which may lead to unstable performance during optimization.

Chapter 4

Application in Deep Latent Variable Models

In this chapter, we apply HEI-KSD/maxSKSD to improve the performance of Deep Latent Variable Models, a class of Deep Generative Models which model complex data using neural networks. In particular, we explore whether we can train improved probabilistic model using HEI-KSD and HEI-maxSKSD training scheme, compared with model trained using vanilla HEI training scheme.

4.1 Deep Latent Variable Model

Deep Latent Variable Model (DLVM) assumes that each observation in our dataset $\mathbf{x}_n \in R^D$ can be generated from a corresponding latent variable $\mathbf{z}_n \in R^K$ (K is usually much less than D). The generative process assumed is that we first sample $\mathbf{z}_n$ from a prior distribution $p(\mathbf{z}_n)$ and then generate $\mathbf{x}_n$ according to the conditional distribution $p(\mathbf{x}_n|\mathbf{z}_n)$, which we call the likelihood function. Usually the prior is assumed to be a standard Gaussian distribution ($p(\mathbf{z}_n) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$) and the likelihood function is modelled using a neural network with parameter θ. Our goal is to find the set of parameters θ^* that maximizes the marginal likelihood (or partition) of the dataset $\{\mathbf{x}_n\}_{n=1}^N$:

$$\begin{aligned}\mathcal{Z} = p(\{\mathbf{x}_n\}_{n=1}^N) &= \int p(\{\mathbf{z}_n\}_{n=1}^N) p_\theta(\{\mathbf{x}_n\}_{n=1}^N | \{\mathbf{z}_n\}_{n=1}^N) d\mathbf{z}_{1:N} \\ &= \prod_{n=1}^N \int p(\mathbf{z}_n) p_\theta(\mathbf{x}_n|\mathbf{z}_n) d\mathbf{z}_n\end{aligned} \tag{4.1}$$

To avoid numerical issue, in practice we instead try to maximize the log marginal likelihood (or log-partition) of the data: $\log \mathcal{Z} = \sum_{n=1}^N \log \int p(\mathbf{z}_n) p_\theta(\mathbf{x}_n|\mathbf{z}_n) d\mathbf{z}_n$. However, the log-partition is intractable since it involves complicated integrals with respect to latent variables, which often live in high dimension.

4.1.1 Variational Autoencoder

To address this intractability, [20] proposed to use Variational Inference to define a variational distribution $q_\phi(\mathbf{z}_n|\mathbf{x}_n)$ with parameters ϕ, to approximate the true posterior distribution $p(\mathbf{z}_n|\mathbf{x}_n)$, which is the target distribution. They call the method Variational Autoencoder (VAE). In particular, they used a factorized Gaussian distribution to model the true posterior. For each observation $\mathbf{x}_n$, the mean and the diagonal covariance matrix of $q_\phi(\mathbf{z}_n|\mathbf{x}_n)$ are obtained using another neural network with parameters ϕ. With this approximate posterior distribution of latent variable, we can then obtain the evidence lower bound of the log-partition of dataset:

$$L_{ELBO} = \frac{1}{N}\sum_{n=1}^{N} E_{q_\phi(\mathbf{z}_n|\mathbf{x}_n)}[\log p(\mathbf{x}_n, \mathbf{z}_n) - \log q_\phi(\mathbf{z}_n|\mathbf{x}_n)] \tag{4.2}$$

Unlike the true posterior which involves the intractable partition in the denominator, the unnormalized true posterior $p(\mathbf{x}_n, \mathbf{z}_n) = p(\mathbf{z}_n)p_\theta(\mathbf{x}_n|\mathbf{z}_n)$ can be computed analytically, and since it is easy to generate samples from the factorized Gaussian distribution, a Monte Carlo estimate of L_{ELBO}, $\hat{L}_{ELBO}$, can be easily computed. In practice, θ and ϕ can be trained jointly by maximizing $\hat{L}_{ELBO}$ using Stochastic Gradient Descent based optimization algorithm and the reparametrization trick [20] is applied to generate samples from $q_\phi(\mathbf{z}_n|\mathbf{x}_n)$ in order to take advantage of the auto differentiation technique.

As discussed in Chapter 1, the approximation bias of VAE is inevitable unless the true posterior does follow a factorized Gaussian distribution, which is unrealistic. The performance of VAE highly depends on how well $q_\phi[\mathbf{z}_n|\mathbf{x}_n]$ models the true posterior distribution.

4.1.2 Importance Weighted Autoencoder (IWAE)

[32] proposed to improve the performance of VAE using a tighter lower bound to the log-partition of the dataset as the training objective: the k-sample importance weighting estimate of $\log p(\{\mathbf{x_n}\}_{n=1}^{N})$, which is given below.

$$L^k_{IWAE} = \frac{1}{N}\sum_{n=1}^{N} E_{q_\phi(\mathbf{z}_n|\mathbf{x}_n)}[\log \frac{1}{k}\sum_{i=1}^{k} \frac{p_\theta(\mathbf{x}_n, \mathbf{z}^i_n)}{q_\phi(\mathbf{z}^i_n|\mathbf{x}_n)}] \tag{4.3}$$

It can be shown that L^k_{IWAE} is a lower bound to the log-partition and the tightness of the bound improves as more samples are used. VAE can be seen as a special case of IWAE: when $k = 1$, L_{ELBO} is recovered. It is worth noting that maximizing L^k_{IWAE} is equivalent to minimizing α-divergence with $\alpha = 1$ between $q_\phi(\mathbf{z}_{1:N}|\mathbf{x}_{1:N})$ and $p(\mathbf{z}_{1:N}|\mathbf{x}_{1:N})$ ([8]). In practice, we use Doubly Reparametrized IWAE objective DReG-L^k_{IWAE} [19] to reduce the variance in the gradient estimate of the IWAE objective:

$$\text{DReG-}\hat{L}^k_{IWAE} = \frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{k}[(\frac{w_n^i}{\sum_{j=1}^k w_n^j})^2 \log w_n^i] \tag{4.4}$$

where $w_n^i = \frac{p_\theta(\mathbf{x}_n, \mathbf{z}_n^i)}{q_\phi(\mathbf{z}_n^i|\mathbf{x}_n)}$ and the term $(\frac{w_n^i}{\sum_{j=1}^k w_n^j})^2$ in front of $\log w_n^i$ is treated as constant during optimization. Note that this objective is the same as DReG-$L_{\alpha=1}$ shown in Section 2.1.

Since training IWAE is equivalent to minimizing α-divergence with $\alpha = 1$, IWAE may prefer more dispersed approximate posterior than VAE, in which maximizing L_{ELBO} is equivalent to minimizing α-divergence with $\alpha = 0$.

4.1.3 Training DLVM with HEI-KSD and HEI-maxSKSD

We again can use HMC to reduce the approximation bias of VAE or IWAE and thus improve the performance of DLVM. In particular, for each observation $\mathbf{x}_n$ in the dataset, HEI takes its approximate posterior in VAE or IWAE model, $q_\phi(\mathbf{z}_n|\mathbf{x}_n)$, as initial distribution q_0 and generates initial samples from the initial distribution, then it runs multiple HMC for T transitions with the unnormalized target π^* being the unnormalized true posterior distribution $p_\theta(\mathbf{x}_n, \mathbf{z}_n) = p(\mathbf{z}_n)p_\theta(\mathbf{x}_n|\mathbf{z}_n)$. The final states of HMC (q_T) can then be used to approximate the true posterior. The EI objective for training HMC hyperparamters ψ in this case can be written as follows:

$$L_{EI} = \frac{1}{N}\sum_{n=1}^{N} E_{q_T(\mathbf{z}_n|\mathbf{x}_n;\phi,\theta,\psi)}[\log p_\theta(\mathbf{x}_n, \mathbf{z}_n)] \tag{4.5}$$

Unlike VAE or IWAE in which the model parameters θ (decoder/generative network parameters) are trained along with the encoder/inferential network parameters ϕ, the model (decoder/generative network) parameters θ now are also trained to maximize L_{EI} along with the HMC hyperparameters ψ, since if we trained θ along with ϕ by optimizing L_{ELBO} or DReG-L^k_{IWAE}, the optimal model parameters $\hat{\theta}$ will be the same as that obtained in VAE or IWAE. In conclusion, the training scheme of HEI applied to DLVM can be summarized as follows:

$$\begin{aligned} &\max_\phi L_{ELBO} \text{ or } \max_\phi \text{DReG-}L^k_{IWAE} \\ &\max_{\psi,\theta} L_{EI} \end{aligned} \tag{4.6}$$

As discussed previously, optimizing L_{EI} can result in mode collapse pathology if the initial distribution has small entropy, which is detrimental to the performance of the model since if the approximate distribution $q_T(\mathbf{z}_n|\mathbf{x}_n; \phi, \theta, \psi)$ does not match the target $p_\theta(\mathbf{z}_n|\mathbf{x}_n)$ well, $p_\theta(\mathbf{z}_n|\mathbf{x}_n)$ may in turn try to fit the poor approximate distribution $q_T(\mathbf{z}_n|\mathbf{x}_n; \phi, \theta, \psi)$ during optimization, resulting in poorly trained model parameters. We deduce that HEI with initial distribution q_0 trained by optimizing L^k_{IWAE} ($\alpha = 1$) may suffer less from the mode collapse pathology than HEI with q_0 trained by optimizing L_{ELBO} ($\alpha = 0$), since the initial distribution in the former HEI tends to have larger width. However, as we see in the 2D experiments, the entropy of the initial distribution trained by optimizing $L_{\alpha=1}$ may still not be enough to avoid the pathology for some distributions. Thus, we again incorporate an inflation parameter s into our model so that the mean of the initial distribution is the same as that in vanilla HEI, but the marginal variances of the initial distribution is obtained by multiplying the marginal variances of the approximate distribution in vanilla HEI by s. s is again trained by minimizing KSD or maxSKSD between $q_T(\mathbf{z}_n|\mathbf{x}_n; \phi, \theta, \psi, s)$ and the unnormalized target $\pi^* = p_\theta(\mathbf{z}_n, \mathbf{x}_n)$. The resulting method is called HEI-KSD or HEI-maxSKSD. We hope the entropy of the initial distribution of HEI-KSD/maxSKSD is large enough to avoid the pathology but also is not too large to ruin the convergence of HMC. The training scheme of HEI-KSD/maxSKSD is summarized as follows:

$$\begin{aligned}
&\max_\phi L_{ELBO} \text{ or } \max_\phi \text{DReG-}L^k_{IWAE}\\
&\max_{\psi,\theta} L_{EI}\\
&\min_s S^2(q_T(\mathbf{z}_n|\mathbf{x}_n; \phi, \theta, \psi, s), p_\theta(\mathbf{z}_n, \mathbf{x}_n)) \text{ or } \min_s SK_{max}(q_T(\mathbf{z}_n|\mathbf{x}_n; \phi, \theta, \psi, s), p_\theta(\mathbf{z}_n, \mathbf{x}_n))
\end{aligned} \tag{4.7}$$

where S^2 and SK_{max} represent KSD and maxSKSD respectively.

4.2 Setup of Experiments

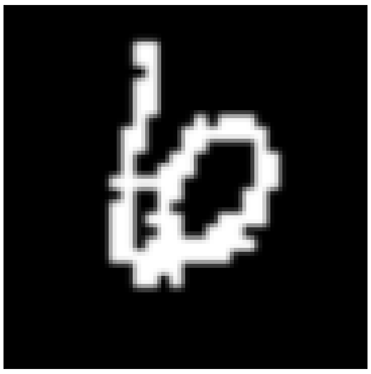


Figure 4.1: Example of image of binzrized MNIST

We consider training HEI-KSD and HEI-maxSKSD on dynamically binarized MNIST training set, which contains 60000 observations. Each observation is a binarized image of handwritten digit with 28×28 pixels and each pixel is either white or black. Figure 4.1 shows an example of such handwritten digit. The likelihood $p_\theta(\mathbf{x}_n|\mathbf{z}_n)$ (decoder) is assumed to be a Bernoulli distribution over all $28 \times 28 = 784$ pixels, and it is modelled by a neural network which outputs the probability that each pixel is white. The approximate posterior $p_{\theta,\psi,\phi,s}(\mathbf{z}_n|\mathbf{x}_n)$ is modelled by a neural network with parameters ϕ (encoder) followed by a finite-length HMC with hyperparameters ψ. This neural network trained by optimizing L_{ELBO} or L^k_{IWAE} outputs the variational mean and covariance for variational factorized Gaussian and then the initial distribution q_0 for HMC is the factorized Gaussian whose marginial variances are obtained by multiplying the variational marginal variances by the inflation parameter s. The mean of this factorized Gaussian is still the variational mean. Then we pass the samples generated from q_0 to the HMC, which outputs the final samples ($q_T = p_{\theta,\psi,\phi,s}(\mathbf{z}_n|\mathbf{x}_n)$) that can be used to approximate the posterior.

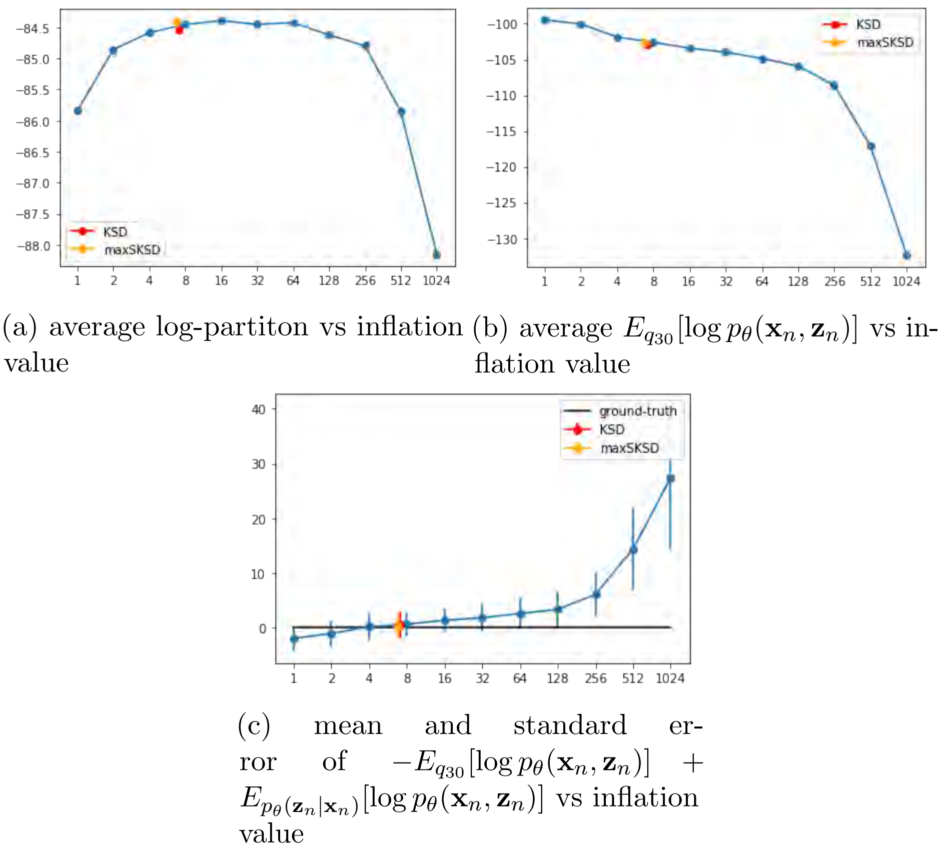
We use HMC with length $T = 30$ and we only consider optimizing its step sizes which are assumed to be different for each sample dimension and for each HMC iteration. The momentum variances are fixed to be 1 for all sample dimensions and for all HMC iterations. We follow similar architecture of the two neural networks (encoder and decoder) used in [6] and the detailed architecture is included in Appendix C. We set the dimension of the latent variable $\mathbf{z}_n$ to be 32. we use Adam ($\beta_1 = 0.9, \beta_2 = 0.99, \epsilon = 10^{-8}$) as the optimization algorithm with initial learning rate being 0.0002, which decays exponentially every 1000 updates with rate 0.97. We train HEI-KSD/maxSKSD for 50000 updates with mini-batch size being 128. Each HMC step size is initialized randomly according to a uniform distribution over $(0.03, 0.06)$ and we force each step size to be greater than 0.01 during optimization. The encoder and decoder parameters ϕ, θ are initialized through pre-training. We perform standard VAE or DReG-IWAE training on ϕ and θ by maximizing L_{ELBO} or L^k_{IWAE} for 100000 updates before we switch to HEI-KSD/maxSKSD training. The pre-training is much less computationally expensive than training HEI-KSD/maxSKSD since it does not involve sampling from HMC. Thus it is worth initializing θ and ϕ with reasonable values through pre-training at a small additional cost. For each update, we use 1 sample and 30 samples for each observation in the mini-batch to estimate L_{EI} and KSD/maxSKSD respectively. For update of ϕ, if we train it by maximizing L_{ELBO} ($\alpha = 0$), then we use 1 sample. Alternatively, if we train ϕ by maximizing L^k_{IWAE} ($\alpha = 1$), then we use $k = 5$ samples to estimate L^5_{IWAE}.

To evaluate the performances of our models, we consider test them on the 10000 prebinarised MNIST test images used in [32] and we use Hamiltonian Annealed Importance Sampling (HAIS) [33] to estimate the ground-truth log marginal likelihood of the prebinarized test set:

$$\mathcal{Z} = \prod_{n=1}^{10000} \int p(\mathbf{z}_n) p_{\theta^*}(\mathbf{x}_n|\mathbf{z}_n) d\mathbf{z}_n \tag{4.8}$$

where $\{\mathbf{x}_n\}_{n=1}^{10000}$ are observations in the test set and θ^* represents the trained model parameters (decoder parameters). We use 1000 annealed steps and 5 leapfrog iterations per annealed step and we tune the HAIS step size to keep acceptance ratio around 65%.

4.3 Experimental Results



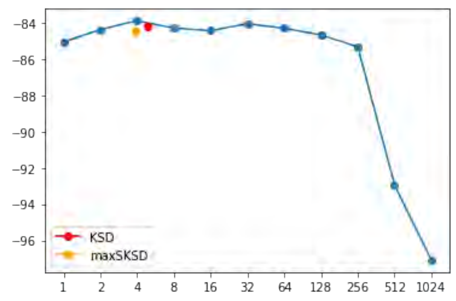
(a) average log-partiton vs inflation value (b) average $E_{q_{30}}[\log p_\theta(\mathbf{x}_n, \mathbf{z}_n)]$ vs inflation value

(c) mean and standard error of $-E_{q_{30}}[\log p_\theta(\mathbf{x}_n, \mathbf{z}_n)] + E_{p_\theta(\mathbf{z}_n|\mathbf{x}_n)}[\log p_\theta(\mathbf{x}_n, \mathbf{z}_n)]$ vs inflation value

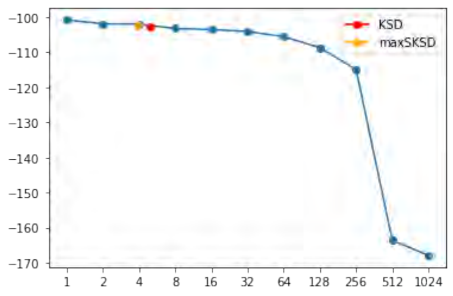
Figure 4.2: Performance of HEI based models vs values of inflation parameters. $\alpha = 0$

To see whether HEI-KSD or HEI-maxSKSD can find an inflation parameter with proper value so that the performance is improved, we first compare performances of HEI-KSD or HEI-maxSKSD models with performances of HEI models with different fixed inflation parameters, that is the initial distribution has the same mean as variational distribution but has the marginal variances that are obtained by multiplying the marginal variances of variational distribution by fixed inflation parameter. We consider fixed inflation parameters from the following set $\{2^x; 0 \leq x \leq 10 \text{ and } x \in \mathcal{Z}\}$. Figure 4.2 plots the performances (average log-partition ($\log \mathcal{Z}$) over 200 randomly sampled images from test set, estimated by HAIS with 100 parallel chains for each image) of different models (HEI with different fixed inflation parameters, HEI-KSD, HEI-maxSKSD) with variational distribution trained by maximizing L_{ELBO} ($\alpha = 0$). It also plots $\frac{1}{200}\sum_{n=1}^{200} E_{q30(\mathbf{z}_n|\mathbf{x}_n;\theta,\phi,\psi,s)}[\log p_\theta(\mathbf{x}_n, \mathbf{z}_n)]$ for these models, where $\{\mathbf{x}_n\}_{n=1}^{200}$ are the 200 randomly chosen images from the test set. Moreover, to examine the convergence of these models, it also plots the empirical mean and standard error of $-E_{q30(\mathbf{z}_n|\mathbf{x}_n;\theta,\phi,\psi,s)}[\log p_\theta(\mathbf{x}_n, \mathbf{z}_n)] + E_{p_\theta(\mathbf{z}_n|\mathbf{x}_n)}[\log p_\theta(\mathbf{x}_n, \mathbf{z}_n)]$ over 200 randomly chosen test images, where $\pi(\mathbf{z}_n) = p_\theta(\mathbf{z}_n|\mathbf{x}_n)$ are approximated by HAIS. Figure 4.3 shows same plots for models with variational distribution trained by maximizing $L_{IWAE}^{k=5}$ ($\alpha = 1$).

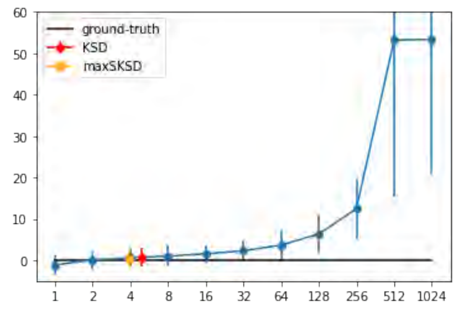
From Figure 4.2 (a) and 4.3 (a), we can see that with moderate inflation values (roughly



(a) average log-partiton vs inflation value



(b) average $E_{q30}[\log p_\theta(\mathbf{x}_n, \mathbf{z}_n)]$ vs inflation value



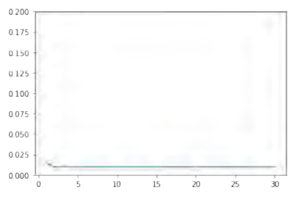
(c) mean and standard error of $-E_{q30}[\log p_\theta(\mathbf{x}_n, \mathbf{z}_n)] + E_{p_\theta(\mathbf{z}_n|\mathbf{x}_n)}[\log p_\theta(\mathbf{x}_n, \mathbf{z}_n)]$ vs inflation value

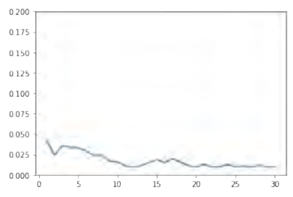
Figure 4.3: Performance of HEI based models vs values of inflation parameters. $\alpha = 1$

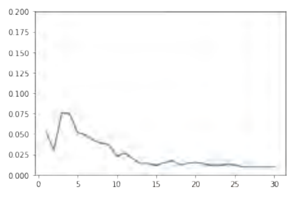
4 to 126 for $\alpha = 0$ and 4 to 32 for $\alpha = 1$), the performances of HEI models are comparable and are better than HEI models trained with too small or too large fixed inflation values. No matter which method is used to train the variational approximation of the target ($\alpha = 0$ or $\alpha = 1$), the optimal inflation values found by HEI-KSD or HEI-maxSKSD fall in the range of inflation values which result in improved model performance and they are close to each other. Plot (b) from two figures show that with inflation values increasing, $\frac{1}{200}\sum_{n=1}^{200} E_{q30(\mathbf{z}_n|\mathbf{x}_n;\theta,\phi,\psi,s)}[\log p_\theta(\mathbf{x}_n, \mathbf{z}_n)]$ keeps decreasing, even when the performance (log-partition) is in fact improved. It suggests $E_{q30(\mathbf{z}_n|\mathbf{x}_n;\theta,\phi,\psi,s)}[\log p_\theta(\mathbf{x}_n, \mathbf{z}_n)]$ is

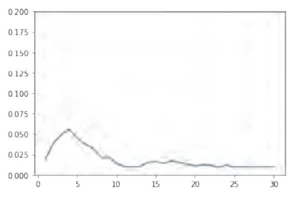
not a good metric to look at for model evaluation. However, it can give us some clue of the approximate posterior. With too small inflation, $\frac{1}{200}\sum_{n=1}^{200} E_{q_{30}(\mathbf{z}_n|\mathbf{x}_n;\theta,\phi,\psi,s)}[\log p_\theta(\mathbf{x}_n, \mathbf{z}_n)]$ is high, suggesting the approximate posterior suffers from mode collapse. On the other hand, if the inflation is too large, $\frac{1}{200}\sum_{n=1}^{200} E_{q_{30}(\mathbf{z}_n|\mathbf{x}_n;\theta,\phi,\psi,s)}[\log p_\theta(\mathbf{x}_n, \mathbf{z}_n)]$ drops significantly, suggesting the finite length HMC is unable to move to the high probability region of the target due to overdispersed initial distribution and thus the approximate posterior has overly large uncertainty. In plot (c), HEI-KSD and HEI-maxSKSD demonstrate good convergence since the mean of $-E_{q_{30}(\mathbf{z}_n|\mathbf{x}_n;\theta,\phi,\psi,s)}[\log p_\theta(\mathbf{x}_n, \mathbf{z}_n)] + E_{p_\theta(\mathbf{z}_n|\mathbf{x}_n)}[\log p_\theta(\mathbf{x}_n, \mathbf{z}_n)]$ over 200 test images is close to 0 and the corresponding standard error is small. The pattern seen in plot (c) is consistent with what we observe in plot (b): HEIs with small fixed inflation values seem to underestimate the uncertainty of the target (mode collapse) while HEIs with large inflation values seem to overestimate the uncertainty of the target. Note that the log-partition of HEIs with inflation values that seem to result in moderate levels of overestimation of target uncertainty can still be good, suggesting the robustness of the decoder is improved with moderately large inflation values and it holds true even when the inflation values are larger than that required for convergence. It is also worth noting that the performance (log-partition) and convergence of vanilla HEI (with fixed inflation 1) with variational distribution tuned by maximizing DReG-$LIWAE^{k=5}$ are both better than vanilla HEI with variational distribution tuned by maximizing L_{ELBO}, suggesting α-divergence minimization with $\alpha = 1$ results in approximate distribution with larger width than α-divergence minimization with $\alpha = 0$, which may explain why the optimal inflation values found by HEI-KSD and HEI-maxSKSD with variational distribution trained by maximizing L_{ELBO} are larger than those found by HEI-KSD and HEI-maxSKSD with variational distribution trained by maximizing $L_{IWAE}^{k=5}$.

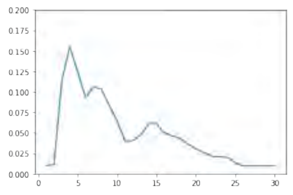
To have a closer look at how HMC behaves for each model, in Figure 4.4, we plot HMC step size against HMC iteration number (30 iterations in total), for models shown in Figure 4.2, and for 1 out of 32 latent dimensions. The models that have high performances, including the models with inflation tuned by optimizing KSD or maxSKSD, have moderate inflation values ($4 = 2^2$ to $128 = 2^7$). We can see that the behaviours of step sizes of these good models are similar: the chains take large global steps for a few initial iterations to move to the high target probability regions, and then make smaller local moves. This may improve the robustness of the model trained as the HMC chains won't always generate samples close to the mode of the target, instead they may provide a variety of different latent variables such that the decoder has to learn to deal with not just one, but multiple latent representations of the data. For models with too small inflation values, the HMC chains are lack of exploration of the target. In particular, the step sizes of vanilla HEI ($s = 1$) are all close to 0.01 which is the minimum value we force the step sizes to be greater than, suggesting that vanilla HEI has poor mixing and it prefers distribution that is close to the initial distribution, which is the variational approximation of the target. We deduce that this insufficiency of exploration will be more severe if we do not force step sizes to be greater than 0.01 during optimization. On the contrary, for too large inflation values, the initial distributions are too different from the target and it is very hard for HMC with just 30 steps to converge to the target. Thus, most step sizes tend to be much larger than those in well performed models, suggesting HMCs are making every effort to return to the high probability regions of the target.

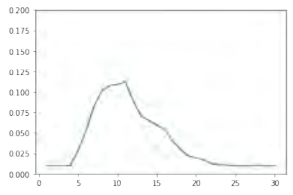

(a) $s = 2^0$

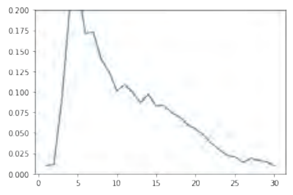

(b) $s = 2^1$

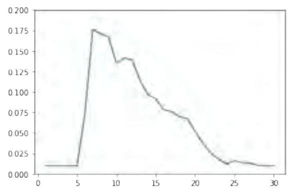

(c) $s = 2^2$

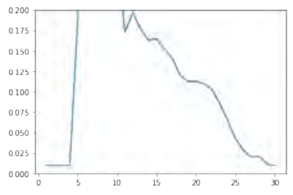

(d) $s = 2^3$

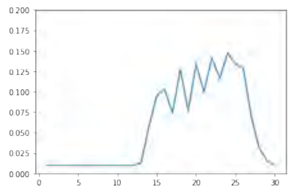

(e) $s = 2^4$

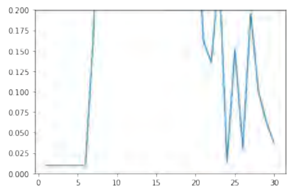

(f) $s = 2^5$

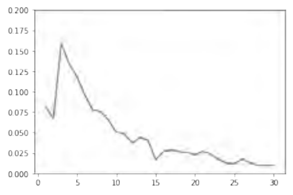

(g) $s = 2^6$


(h) $s = 2^7$


(i) $s = 2^8$


(j) $s = 2^9$


(k) $s = 2^{10}$


(l) ksd: $s = 7.0586$

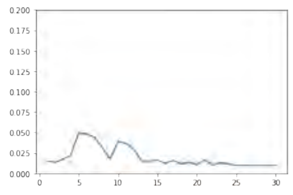

(m) maxsksd: $s = 6.7855$

Figure 4.4: Step size vs HMC layer number for difference inflation values, shown for only one dimension out of the 32 latent dimensions. $\alpha = 0$

Table 4.1 shows average log-partition estimated by HAIS over the whole test set for different models. We can see that vanilla HEI outperforms VAE and IWAE, suggesting VAE and IWAE can be improved by further running HMC tuned by L_{EI} since the resulting approximate distributions are closer to the target than variational approximations. For vanilla HEI, tuning initial distribution by maximizing $L_{IWAE}^{k=5}$ (vanilla HEI ($\alpha = 1$)) gives us better result than maximizing L_{ELBO} (vanilla HEI ($\alpha = 0$)), suggesting its initial distribution has larger width. However, after using KSD or maxSKSD to tune the inflation parameter, the performances of models are comparable and are better than vanilla HEI no matter which method is used to tune the variational distribution, suggesting the proper inflation parameter contributes most to improving the performance. Like in 2D experiments, we also tried to tune HMC hyperparameters ψ along with inflation s using KSD or maxSKSD instead of L_{EI} (the variational distribution is tuned by maximizing L_{ELBO} $(\alpha = 0)$) but it leads to divergence, suggesting optimization using KSD or maxSKSD as objective to tune many parameters may be unstable. We also used pretrained parameters from HEI-KSD or HEI-maxSKSD as a proper initialization and then switched to using KSD or maxSKSD to train ψ and s, but it still leads to divergence, which further shows the advantage of L_{EI} against KSD or maxSKSD in training ψ. However, with pretrained initialization, the model using maxSKSD to train ψ and s has better performance than that uses KSD, suggesting maxSKSD may mitigate the curse-of-dimensionality problem of KSD to some extent.

	log-partition
VAE $(\alpha = 0)$	-85.1022
DReG-IWAE $(\alpha = 1)$	-83.7528
vanilla HEI $(\alpha = 0)$	-83.4882
vanilla HEI $(\alpha = 1)$	-82.4565
HEI-KSD $(\alpha = 0)$	-81.9872
HEI-maxSKSD $(\alpha = 0)$	-81.9176
HEI-KSD $(\alpha = 1)$	**-81.8755**
HEI-maxSKSD $(\alpha = 1)$	-82.0090
KSD $(\psi{+}s)$, $\alpha = 0$	-193.6095
maxSKSD $(\psi{+}s)$, $\alpha = 0$	-261.5302
KSD $(\psi{+}s)$, $\alpha = 0$, pretrain	-226.1570
maxSKSD $(\psi{+}s)$, $\alpha = 0$, pretrain	-125.0672

Table 4.1: Log marginal likelihoods of the whole test set for different models

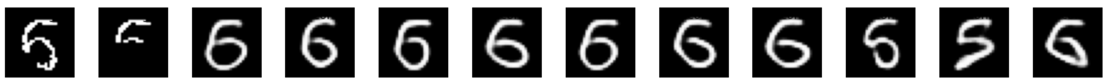
4.4 Imputation Task

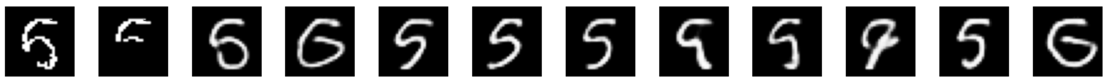
In addition to estimating the average test set log-partition, we also carried out an imputation task, similar to the one in [16], for model evaluation. We first randomly selected 100 test images and removed the pixels in the lower half of the images. Then we imputed the values of missing pixels by (approximate) posterior sampling from the DLVM models. We computed the label accuracy for the imputed test images (detailed setup can be found in Appendix D). A good model should give imputed images that are similar to the original images. Thus, it should achieve high label accuracy. Table 4.2 shows the label accuracy of the imputed images for different models. DReG-IWAE has considerably

	label accuracy
VAE ($\alpha = 0$)	0.7694
DReG-IWAE ($\alpha = 1$)	0.6805
vanilla HEI ($\alpha = 0$)	0.7984
HEI-KSD ($\alpha = 0$)	0.8151
HEI-maxSKSD ($\alpha = 0$)	0.8157
vanilla HEI ($\alpha = 1$)	0.8133
HEI-KSD ($\alpha = 1$)	0.8179
HEI-maSKSD ($\alpha = 1$)	**0.8287**

Table 4.2: Label accuracy for imputed images

lower label accuracy than the other models. It may suggest inferential network (encoder) of DReG-IWAE is poorly trained compared with the other models, so that it is easy to generate samples around wrong modes. The label accuracy of VAE is noticeably worse than HEI based models, suggesting its approximate posterior samples are also often generated around wrong modes. For $\alpha = 0$ or $\alpha = 1$, HEI-KSD and HEI-maxSKSD show better performance than vanilla HEI since the label accuracy of HEI-KSD and HEI-maxSKSD are both higher than vanilla HEI. For some test images with missing pixels, HEI-KSD and HEI-maxSKSD can achieve considerably higher label accuracy than vanilla HEI and also have approximate posteriors that are able to capture the multi-modality nature of the true posterior. Figure 4.5 shows an example of this phenomenon. We can see that for vanilla HEI the imputed images generated are stuck around a wrong mode. However, for HEI-KSD and HEI-maxSKSD, the imputed images generated are mostly correct and they also demonstrate higher imputation diversity than vanilla HEI.


imputed images from vanilla HEI ($\alpha = 0$)


imputed images from HEI-KSD ($\alpha = 0$)

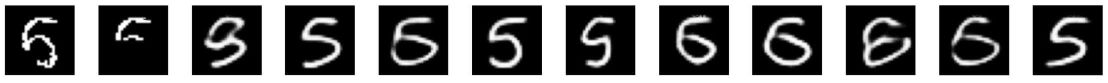

imputed images from HEI-maxSKSD ($\alpha = 0$)

Figure 4.5: Imputed images for a test example from different models. The first two images in each subfigure are the original image and image with lower half removed respectively

Chapter 5

Conclusions and Future Work

This chapter summarizes conclusions made in this work and proposes some possible future work.

In Chapter 1, we introduced vanilla HEI and relevant concepts. We also demonstrated the case where HEI fails due to mode collapse pathology and showed that this pathology can be overcome by using initial distribution with sufficiently large entropy. In Chapter 2 we proposed two methods to find the initial distribution with enough entropy to overcome the pathology. One is to tune the initial distribution parameters using α-divergence minimization with $\alpha = 1$ so that the initial distribution can have larger width than that obtained by VI. The other method is to use initial distribution with same mean as the variational distribution but with marginal variances obtained by multiplying the marginal variances of the variational distribution by an inflation parameter s, which is tuned by KSD or maxSKSD minimization. We call these models HEI-KSD or HEI-maxSKSD. In chapter 3 and 4, we applied HEI-KSD and HEI-maxSKSD to sample from 2D densities and to train DLVM respectively. We found that for these two applications, HEI-KSD and HEI-maxSKSD can successfully overcome the mode collapse pathology of HEI and achieve better performance.

KSD suffers from curse-of-dimensionality. Possible future work includes applying HEI-KSD and HEI-maxSKSD in higher dimension problems than considered in this work. For example, it would be interesting to apply HEI-KSD and HEI-maxSKSD to fit Bayesian Neural Network, in which the target distribution is the posterior of the network weights. Then we can investigate whether HEI-KSD will be affected by curse-of-dimensionality and whether HEI-maxSKSD can show advantage against HEI-KSD for such high dimensional target. It would also be interesting to investigate the performance of HEI-KSD and HEI-maxSKSD based on kernel different from RBF kernel. For instance, IMQ kernel proposed in [25] may be a good choice.

Bibliography

[1] K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. ISBN: 9780262018029.

[2] M. Jordan, Z. Ghahramani, T. Jaakkola, and Saul L. An introduction to variational methods for graphical models. *Machine Learning*, 1999. 37:183–233.

[3] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, and A. H. Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics 21.6*, 1953. 1087–1092.

[4] R. M. Neal. Bayesian learning for neural networks. *PhD thesis, University of Toronto*, 1996.

[5] R. M. Neal. Mcmc using hamiltonian dynamics. 2010.

[6] Y. Zhang and J. M. Hernández-Lobato. Ergodic inference: Accelerate convergence by optimisation. *arXiv:1805.10377, 2019*, 2019.

[7] C. M. Bishop. *Pattern Recognition and Machine Learning Section 10.1: variational inference*. Springer, 2006. ISBN: 9780387310732.

[8] J. M. Hernández-Lobato, Y. Li, M. Rowland, D. Hernández-Lobato, T. Bui, and R. E. Turner. Black-box α-divergence minimization. *The International Conference on Machine Learning (ICML)*, 2015.

[9] T. Salimans, D. Kingma, and M. Welling. Markov chain monte carlo and variational inference: Bridging the gap. *The International Conference on Machine Learning (ICML)*, 2015.

[10] M. D. Hoffman. Learning deep latent gaussian models with markov chain monte carlo. In *Proceedings of the 34th International Conference on Machine Learning, volume 70 of Proceedings of Machine Learning Research*, page 1510–1519, 2017.

[11] Y. Zhang, J. M. Hernández-Lobato, and Z. Ghahramani. Ergodic measure preserving flows. *arXiv:1805.10377, 2018*, 2018.

[12] T. Cover and J. Thomas. Elements of information theory. *John Wiley & Sons*, 1991.

[13] T. Han, Y. Lu, S. C. Zhu, and Y. N. Wu. Alternating back-propagation for generator network. *AAAI, volume 3*, page 13, 2017.

[14] F. Ruiz and M. Titsias. A contrastive divergence for combining variational inference and mcmc. *The International Conference on Machine Learning (ICML)*, 2019.

[15] Q. Liu, J. D. Lee, and M. Jordan. A kernelized stein discrepancy for goodness-of-fit tests. *The International Conference on Machine Learning (ICML)*, 2016.

[16] W. Gong, Y. Li, and J. M. Hernández-Lobato. Sliced kernelized stein discrepancy. *arXiv:2006.16531, 2020*, 2020.

[17] T. P. Minka. Expectation propagation for approximate bayesian inference. *UAI*, pages 362–369, 2001.

[18] T. Rainforth, A. R. Kosiorek, T. AnhLe, C. J. Maddison, M. Igl, F. Wood, and Y. W. Teh. Tighter variational bounds are not necessarily better. *International Conference on Machine Learning (ICML)*, 2018.

[19] G. Tucker, D. Lawson, S. Gu, and C. J. Maddison. Doubly reparameterized gradient estimators for monte carlo objectives. *International Conference on Machine Learning (ICML)*, 2019.

[20] D. Kingma and M. Welling. Auto-encoding variational bayes. *The International Conference on Learning Representations (ICLR)*, 2014.

[21] G. Roeder, Y. Wu, and D. Duvenaud. Sticking the landing: An asymptotically zero variance gradient estimator for variational inference. *Advances in Neural Information Processing Systems (NIPS)*, 2017.

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

[23] A. Muller. Integral probability metrics and their generating classes of functions. *Ann. Appl. Probab*, pages 29 (2): 429–443, 1997.

[24] J. Gorham and L. Mackey. Measuring sample quality with stein's method. *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[25] J. Gorham and L. Mackey. Measuring sample quality with kernels. *International Conference on Machine Learning (ICML)*, 2017.

[26] K. Chwialkowski, H. Strathmann, and A. Gretton. A kernel test of goodness of fit. *International Conference on Machine Learning (ICML)*, 2016.

[27] T. Hu, Z. Chen, H. Sun, J. Bai, Y. Mao, and G. Cheng. Stein neural sampler. *arXiv:1810.03545, 2018*, 2018.

[28] Q. Liu and Y. Feng. Two methods for wild variational inference. *arXiv:1612.00081*, 2016.

[29] D. Wang and Q. Liu. Learning to draw samples: With application to amortized mle for generative adversarial learning. *arXiv:1611.01722*, 2016.

[30] Q. Liu and D. Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. *Advances in neural information processing systems (NIPS)*, 2016.

[31] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*, 2015.

[32] Y. Burda, Grosse R., and R. Salakhutdinov. Importance weighted autoencoders. *International Conference on Learning Representations (ICLR)*, 2015.

[33] Y. Wu, Y. Burda, G. Roger, and S. Ruslan. On the quantitative analysis of deep belief networksnalysis of decoder-based generative models. *International Conference on Learning Representations (ICLR)*, 2017.

Appendix A

Derivation of DReG-$L_{\alpha=1}$

Let $w_i = \frac{\pi^*(\mathbf{x}_i)}{p_\phi(\mathbf{x}_i)}$ and $\mathbf{x}_{1:K} \sim \prod_i p_\phi(\mathbf{x}_i)$ can be sampled using reparameterization trick: $\mathbf{x}_i = \mathbf{x}(\epsilon_i; \phi)$, then

$$L_{\alpha=1} = E_{\mathbf{x}_{1:K}}[\log \frac{1}{K}\sum_{i=1}^{K} w_i] = E_{\epsilon_{1:K}}[\log \frac{1}{K}\sum_{i=1}^{K} w_i] \tag{A.1}$$

The gradient of $L_{\alpha=1}$ is

$$\begin{aligned}\nabla_\phi E_{\epsilon_{1:K}}[\log \frac{1}{K}\sum_{i=1}^{K} w_i] &= E_{\epsilon_{1:K}}[\sum_{i=1}^{K} \frac{1}{\sum_{j=1}^{K} w_j}\nabla_\phi w_i] \\ &= E_{\epsilon_{1:K}}[\sum_{i=1}^{K} \frac{w_i}{\sum_{j=1}^{K} w_j}(-\nabla_\phi \log p_\phi(\mathbf{x}_i) + (\nabla_{\mathbf{x}_i} \log w_i)(\nabla_\phi \mathbf{x}_i))]\end{aligned} \tag{A.2}$$

Now we would like to rewrite the term $E_{\epsilon_{1:K}}[\sum_{i=1}^{K} \frac{w_i}{\sum_{j=1}^{K} w_j}\nabla_\phi \log p_\phi(\mathbf{x}_i)]$ to a form with less variance.

$$\begin{aligned}E_{\epsilon_{1:K}}[\sum_{i=1}^{K} \frac{w_i}{\sum_{j=1}^{K} w_j}\nabla_\phi \log p_\phi(\mathbf{x}_i)] &= \sum_{i=1}^{K} E_{\epsilon_{1:K}}[\frac{w_i}{\sum_{j=1}^{K} w_j}\nabla_\phi \log p_\phi(\mathbf{x}_i)] \\ &= \sum_{i=1}^{K} E_{\mathbf{x}_{1:K}}[\frac{w_i}{\sum_{j=1}^{K} w_j}\nabla_\phi \log p_\phi(\mathbf{x}_i)] \\ &= \sum_{i=1}^{K} E_{\mathbf{x}_{-i}\mathbf{x}_i}[\frac{w_i}{\sum_{j=1}^{K} w_j}\nabla_\phi \log p_\phi(\mathbf{x}_i)]\end{aligned} \tag{A.3}$$

where $\mathbf{x}_{-i}$ represents $\mathbf{x}_{1:i-1} \cup \mathbf{x}_{i+1:K}$. The equation above can be rewritten by taking advantage of the equivalence between the REINFORCE gradient and the reparameterization trick gradient (proved in [19]):

$$\begin{aligned} E_{\mathbf{x}_i}[\frac{w_i}{\sum_{j=1}^K w_j}\nabla_\phi \log p_\phi(\mathbf{x}_i)] &= E_{\epsilon_i}[\nabla_{\mathbf{x}_i}(\frac{w_i}{\sum_{j=1}^K w_j})\nabla_\phi \mathbf{x}_i] \\ &= E_{\epsilon_i}[(\frac{w_i}{\sum_{j=1}^K w_j} - \frac{w_i^2}{(\sum_{j=1}^K w_j)^2})(\nabla_{\mathbf{x}_i}\log w_i)(\nabla_\phi \mathbf{x}_i)] \end{aligned} \tag{A.4}$$

Plugging A.4 into A.2, we can get

$$\begin{aligned} \nabla_\phi E_{\epsilon_{1:K}}[\log \frac{1}{K}\sum_{i=1}^K w_i] &= \sum_{i=1}^K E_{\epsilon_i}[(-\frac{w_i}{\sum_{j=1}^K w_j} + \frac{w_i^2}{(\sum_{j=1}^K w_j)^2} + \frac{w_i}{\sum_{j=1}^K w_j})(\nabla_{\mathbf{x}_i}\log w_i)(\nabla_\phi \mathbf{x}_i)] \\ &= \sum_{i=1}^K E_{\epsilon_i}[\frac{w_i^2}{(\sum_{j=1}^K w_j)^2}(\nabla_{\mathbf{x}_i}\log w_i)(\nabla_\phi \mathbf{x}_i)] \\ &= E_{\epsilon_{1:K}}[\sum_{i=1}^K(\frac{w_i}{\sum_{j=1}^K w_j})^2(\nabla_{\mathbf{x}_i}\log w_i)(\nabla_\phi \mathbf{x}_i)] \end{aligned} \tag{A.5}$$

Therefore, the gradient of DReG-$L_{\alpha=1}$ with respect to ϕ is $E_{\epsilon_{1:K}}[\sum_{i=1}^K(\frac{w_i}{\sum_{j=1}^K w_j})^2(\nabla_{\mathbf{x}_i}\log w_i)(\nabla_\phi \mathbf{x}_i)]$.

Appendix B

2D densities used in Chapter 3

Table B.1 shows the log-unnormalized-densities of the distributions used in chapter 3.

Label	Name	unnormlized log-density ($\log \pi^*(\mathbf{x})$)
i	Laplace	$-\lvert x_1 - 5\rvert - \lvert x_2 - 5\rvert$
ii	Dual Moon	$-3.125(\sqrt{x_1^2 + x_2^2} - 2)^2 + \log[e^{-0.5(\frac{x_1+2}{0.6})^2} + e^{-0.5(\frac{x_1-2}{0.6})^2}]$
iii	Gaussian Mixture	$\log[\sum_{i=1}^{7} e^{-0.5[(x_1-5\cos(\frac{2i\pi}{7}))^2+(x_2-5\sin(\frac{2i\pi}{7}))^2]}]$
iv	Wave1	$-0.5(\frac{x_2+\sin(0.5\pi x_1)}{0.4})^2$
v	Wave2	$e^{-0.5(\frac{x_2+\sin(0.5\pi x_1)}{0.35})^2} + e^{-0.5(\frac{-x_2-\sin(0.5\pi x_1)+3e^{-\frac{0.5}{0.36}(x_1-1)^2}}{0.35})^2}$
vi	Wave3	$e^{-0.5(\frac{x_2+\sin(0.5\pi x_1)}{0.4})^2} + e^{-0.5(\frac{-x_2-\sin(0.5\pi x_1)+\frac{3}{1+e^{-\frac{x_1-1}{0.3}}}}{0.35})^2}$

Table B.1: log-unnormalized-densities used in chapter 3. State space is $\mathbf{x} = [x_1, x_2]^T$

Appendix C

Network Architecture Used in Chapter 4

The dimension of latent variables considered in this work is 32. We follow similar encoder and decoder network architecture as that used in [6]. The encoder network consists of three convolutional layers and 1 fully connected multilayer perceptron (MLP). The filters of convolutional layers have width 5 and stride 2 and the three convolutional layers have 16, 32 and 32 channels respectively. The output of the third convolutional layer is converted to a vector with dimension 512 and then it is passed to the MLP. The MLP has one hidden layer with 500 hidden units and an output layer with 64 neurons (32 neurons for the mean of the factorized Gaussian and 32 neurons for the log-variance of the factorized Gaussian). All activation functions are ReLU except for the output layer which uses a linear activation. The decoder consists of a MLP with one hidden layer with 500 hidden units and an output layer with 512 neurons and three deconvolutional layers whose filters have width 5. The intermediate dimensions of the three deconvolutional layers and the output dimension are $(4 \times 4 \times 32)$, $(7 \times 7 \times 32)$, $(14 \times 14 \times 16)$ and $(28 \times 28 \times 1)$ respectively. All activations are again ReLU except for the output layer which used sigmoid activation.

Appendix D

Setup of The Imputation Task in Section 4.4

To carry out the imputation task, we follow [16] which uses approximate Gibbs sampler. In detail, we denote observed and missing pixels as $\mathbf{x}_o$ and $\mathbf{x}_m$ respectively. We use $q_{\phi,\psi,\theta}(\mathbf{z}|\mathbf{x})$ to represent encoder+HMC in HEI based models and $q_\phi(\mathbf{z}|\mathbf{x})$ to represent encoder in VAE or IWAE. The decoder is represented by $p_\theta(\mathbf{x}|\mathbf{z})$. For each sampler, we applies the following procedure repeatedly:

(1) Generate latent samples: $\mathbf{z} \sim q_{\phi,\psi,\theta}(\mathbf{z}|\mathbf{x}_o, \mathbf{x}_m)$ (HEI based models) or $q_\phi(\mathbf{z}|\mathbf{x}_o, \mathbf{x}_m)$ (VAE, IWAE).
(2) Reconstruction: $\mathbf{x}^* \sim p_\theta(\mathbf{x}^*|\mathbf{z})$.
(3) Imputation: $\mathbf{x}_m \leftarrow \mathbf{x}^*_m$.

We maintain 200 parallel samplers for each of the 100 randomly selected test images and for each sampler we repeat the above procedure for 500 iterations. Then we set the imputation label of each imputed image to be the same as its nearest neighbour from the training set.