

---

# Sliced Score Matching: A Scalable Approach to Density and Score Estimation

---

**Yang Song\***  
Stanford University

**Sahaj Garg\***  
Stanford University

**Jiaxin Shi**  
Tsinghua University

**Stefano Ermon**  
Stanford University

## Abstract

Score matching is a popular method for estimating unnormalized statistical models. However, it has been so far limited to simple, shallow models or low-dimensional data, due to the difficulty of computing the Hessian of log-density functions. We show this difficulty can be mitigated by projecting the scores onto random vectors before comparing them. This objective, called sliced score matching, only involves Hessian-vector products, which can be easily implemented using reverse-mode automatic differentiation. Therefore, sliced score matching is amenable to more complex models and higher dimensional data compared to score matching. Theoretically, we prove the consistency and asymptotic normality of sliced score matching estimators. Moreover, we demonstrate that sliced score matching can be used to learn deep score estimators for implicit distributions. In our experiments, we show sliced score matching can learn deep energy-based models effectively, and can produce accurate score estimates for applications such as variational inference with implicit distributions and training Wasserstein Auto-Encoders.

## 1 INTRODUCTION

Score matching (Hyvärinen, 2005) is particularly suitable for learning unnormalized statistical models, such as energy based ones. It is based on minimizing the distance between the derivatives of the log-density functions (a.k.a., *scores*) of the data and model distributions. Unlike maximum likelihood estimation (MLE), the objective of score

matching only depends on the scores, which are oblivious to the (usually) intractable partition functions. However, score matching requires the computation of the diagonal elements of the Hessian of the model’s log-density function. This Hessian trace computation is generally expensive (Martens et al., 2012), requiring a number of forward and backward propagations proportional to the data dimension. This severely limits its applicability to complex models parameterized by deep neural networks, such as deep energy-based models (LeCun et al., 2006; Wenliang et al., 2019).

Several approaches have been proposed to alleviate this difficulty: Kingma & LeCun (2010) propose approximate backpropagation for computing the trace of the Hessian; Martens et al. (2012) develop curvature propagation, a fast stochastic estimator for the trace in score matching; and Vincent (2011) transforms score matching to a denoising problem which avoids second-order derivatives. These methods have achieved some success, but may suffer from one or more of the following problems: inconsistent parameter estimation, large estimation variance, and cumbersome implementation.

To alleviate these problems, we propose sliced score matching, a variant of score matching that can scale to deep unnormalized models and high dimensional data. The key intuition is that instead of directly matching the high-dimensional scores, we match their projections along random directions. Theoretically, we show that under some regularity conditions, sliced score matching is a well-defined statistical estimation criterion that yields consistent and asymptotically normal parameter estimates. Moreover, compared to the methods of Kingma & LeCun (2010) and Martens et al. (2012), whose implementations require customized backpropagation for deep networks, sliced score matching only involves Hessian-vector products, thus can be easily and efficiently implemented in frameworks such as TensorFlow (Abadi et al., 2016) and PyTorch (Adam et al., 2017).

\* Joint first authors. Correspondence to Yang Song <[yangsong@cs.stanford.edu](mailto:yangsong@cs.stanford.edu)> and Stefano Ermon <[ermon@cs.stanford.edu](mailto:ermon@cs.stanford.edu)>.

Beyond training unnormalized models, sliced score matching can also be naturally adapted as an objective for estimating the score function of a data generating distribution (Sasaki et al., 2014; Strathmann et al., 2015) by training a score function model parameterized by deep neural networks. This observation enables many new applications of sliced score matching. For example, we show that it can be used to provide accurate score estimates needed for variational inference with implicit distributions (Huszar, 2017) and learning Wasserstein Auto-Encoders (WAE, Tolstikhin et al. (2018)).

Finally, we evaluate the performance of sliced score matching on learning unnormalized statistical models (density estimation) and estimating score functions of a data generating process (score estimation). For density estimation, experiments on deep kernel exponential families (Wenliang et al., 2019) and NICE flow models (Dinh et al., 2015) show that our method is either more scalable or more accurate than existing score matching variants.

For score estimation, our method improves the performance of variational auto-encoders (VAE) with implicit encoders, and can train WAEs without a discriminator or MMD loss by directly optimizing the KL divergence between aggregated posteriors and the prior. In both situations we outperformed kernel-based score estimators (Li & Turner, 2018; Shi et al., 2018) by achieving better test likelihoods and better sample quality in image generation.

## 2 BACKGROUND

Given i.i.d. samples  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\} \subset \mathbb{R}^D$  from a data distribution  $p_d(\mathbf{x})$ , our task is to learn an unnormalized density,  $\tilde{p}_m(\mathbf{x}; \boldsymbol{\theta})$ , where  $\boldsymbol{\theta}$  is from some parameter space  $\Theta$ . The model's partition function is denoted as  $Z_{\boldsymbol{\theta}}$ , which is assumed to be existent but intractable. Let  $p_m(\mathbf{x}; \boldsymbol{\theta})$  be the normalized density determined by our model, we have

$$p_m(\mathbf{x}; \boldsymbol{\theta}) = \frac{\tilde{p}_m(\mathbf{x}; \boldsymbol{\theta})}{Z_{\boldsymbol{\theta}}}, \quad Z_{\boldsymbol{\theta}} = \int \tilde{p}_m(\mathbf{x}; \boldsymbol{\theta}) d\mathbf{x}.$$

For convenience, we denote the score functions of  $p_m$  and  $p_d$  as  $\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) \triangleq \nabla_{\mathbf{x}} \log p_m(\mathbf{x}; \boldsymbol{\theta})$  and  $\mathbf{s}_d(\mathbf{x}) \triangleq \nabla_{\mathbf{x}} \log p_d(\mathbf{x})$  respectively. Note that since  $\log p_m(\mathbf{x}; \boldsymbol{\theta}) = \log \tilde{p}_m(\mathbf{x}; \boldsymbol{\theta}) - \log Z_{\boldsymbol{\theta}}$ , we immediately conclude that  $\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})$  does not depend on the intractable partition function  $Z_{\boldsymbol{\theta}}$ .

### 2.1 SCORE MATCHING

Learning unnormalized models with maximum likelihood estimation (MLE) can be difficult due to the intractable partition function  $Z_{\boldsymbol{\theta}}$ . To avoid this, score matching (Hyvärinen, 2005) minimizes the Fisher divergence

between  $p_d$  and  $p_m(\cdot; \boldsymbol{\theta})$ , which is defined as

$$L(\boldsymbol{\theta}) \triangleq \frac{1}{2} \mathbb{E}_{p_d} [\|\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) - \mathbf{s}_d(\mathbf{x})\|_2^2]. \quad (1)$$

Since  $\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})$  does not involve  $Z_{\boldsymbol{\theta}}$ , the Fisher divergence does not depend on the intractable partition function. However, Eq. (1) is still not readily usable for learning unnormalized models, as we only have samples and do not have access to the score function of the data  $\mathbf{s}_d(\mathbf{x})$ .

By applying integration by parts, Hyvärinen (2005) shows that  $L(\boldsymbol{\theta})$  can be written as  $L(\boldsymbol{\theta}) = J(\boldsymbol{\theta}) + C$  (*cf.*, Theorem 1 in Hyvärinen (2005)), where

$$J(\boldsymbol{\theta}) \triangleq \mathbb{E}_{p_d} \left[ \text{tr}(\nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})) + \frac{1}{2} \|\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})\|_2^2 \right], \quad (2)$$

$C$  is a constant that does not depend on  $\boldsymbol{\theta}$ ,  $\text{tr}(\cdot)$  denotes the trace of a matrix, and

$$\nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) = \nabla_{\mathbf{x}}^2 \log \tilde{p}_m(\mathbf{x}; \boldsymbol{\theta}) \quad (3)$$

is the Hessian of the log-density function. The constant can be ignored and the following unbiased estimator of the remaining terms is used to train  $\tilde{p}_m(\mathbf{x}; \boldsymbol{\theta})$ :

$$\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N) \triangleq \frac{1}{N} \sum_{i=1}^N \left[ \text{tr}(\nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}_i; \boldsymbol{\theta})) + \frac{1}{2} \|\mathbf{s}_m(\mathbf{x}_i; \boldsymbol{\theta})\|_2^2 \right],$$

where  $\mathbf{x}_1^N$  is a shorthand used throughout the paper for a collection of  $N$  data points  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$  sampled i.i.d. from  $p_d$ , and  $\nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}_i; \boldsymbol{\theta})$  denotes the Hessian of  $\log \tilde{p}_m(\mathbf{x}; \boldsymbol{\theta})$  evaluated at  $\mathbf{x}_i$ .

**Computational Difficulty.** While the score matching objective  $\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N)$  avoids the computation of  $Z_{\boldsymbol{\theta}}$  for unnormalized models, it introduces a new computational difficulty: computing the trace of the Hessian of a log-density function,  $\nabla_{\mathbf{x}}^2 \log \tilde{p}_m$ . A naïve approach of computing the trace of the Hessian requires  $D$  times more backward passes than computing the gradient  $\mathbf{s}_m = \nabla_{\mathbf{x}} \log \tilde{p}_m$  (see Alg. 2 in the appendix). For example, the trace could be computed by applying backpropagation  $D$  times to  $\mathbf{s}_m$  to get each diagonal term of  $\nabla_{\mathbf{x}}^2 \log \tilde{p}_m$  sequentially. In practice,  $D$  can be many thousands, which can render score matching too slow for practical purposes. Moreover, Martens et al. (2012) argues from a theoretical perspective that it is unlikely that there exists an algorithm for computing the diagonal of the Hessian defined by an arbitrary computation graph within a constant number of forward and backward passes.

### 2.2 SCORE ESTIMATION FOR IMPLICIT DISTRIBUTIONS

Besides parameter estimation in unnormalized models, score matching can also be used to estimate scores of

*implicit distributions*, which are distributions that have a tractable sampling process but without a tractable density. For example, the distribution of random samples from the generator of a GAN (Goodfellow et al., 2014) is an implicit distribution. Implicit distributions can arise in many more situations such as the marginal distribution of a non-conjugate model (Sun et al., 2019), and models defined by complex simulation processes (Tran et al., 2017). In many cases learning and inference become intractable due to the need of optimizing an objective that involves the intractable density.

In these cases, directly estimating the score function  $s_q(\mathbf{x}) = \nabla_{\mathbf{x}} \log q_{\theta}(\mathbf{x})$  based on i.i.d. samples from an implicit density  $q_{\theta}(\mathbf{x})$  can be useful. For example, suppose our learning problem involves optimizing the entropy  $H(q_{\theta}(\mathbf{x}))$  of an implicit distribution. This situation is common when dealing with variational free energies (Kingma & Welling, 2014). Suppose  $\mathbf{x} \sim q_{\theta}$  can be reparameterized as  $\mathbf{x} = g_{\theta}(\epsilon)$ , where  $\epsilon$  is a simple random variable independent of  $\theta$ , such as a standard normal, and  $g_{\theta}$  is a deterministic mapping. We can write the gradient of the entropy with respect to  $\theta$  as

$$\begin{aligned}\nabla_{\theta} H(q_{\theta}) &\triangleq -\nabla_{\theta} \mathbb{E}_{q_{\theta}(\mathbf{x})} [\log q_{\theta}(\mathbf{x})] \\ &= -\nabla_{\theta} \mathbb{E}_{p(\epsilon)} [\log q_{\theta}(g_{\theta}(\epsilon))] \\ &= -\mathbb{E}_{p(\epsilon)} [\nabla_{\mathbf{x}} \log q_{\theta}(\mathbf{x})|_{\mathbf{x}=g_{\theta}(\epsilon)} \nabla_{\theta} g_{\theta}(\epsilon)],\end{aligned}$$

where  $\nabla_{\theta} g_{\theta}(\epsilon)$  is usually easy to compute. The score  $\nabla_{\mathbf{x}} \log q_{\theta}(\mathbf{x})$  is intractable but can be approximated by score estimation.

Score matching is an attractive solution for score estimation since (1) naturally serves as an objective to measure the difference between the a trainable score function and the score of a data generating process. We will discuss this in more detail in Section 3.2 and mention some other approaches of score estimation in Section 5.2.

### 3 DENSITY AND SCORE ESTIMATION WITH SLICED SCORE MATCHING

#### 3.1 SLICED SCORE MATCHING

We observe that one dimensional problems are usually much easier to solve than high dimensional ones. Inspired by the idea of Sliced Wasserstein distance (Rabin et al., 2012), we consider projecting  $s_d(\mathbf{x})$  and  $s_m(\mathbf{x}; \theta)$  onto some random direction  $\mathbf{v}$  and propose to compare their average difference along that random direction. More specifically, we consider the following objective as a replacement of the Fisher divergence  $L(\theta)$  in Eq. (1):

$$L(\theta; p_{\mathbf{v}}) \triangleq \frac{1}{2} \mathbb{E}_{p_{\mathbf{v}}} \mathbb{E}_{p_d} [(\mathbf{v}^T s_m(\mathbf{x}; \theta) - \mathbf{v}^T s_d(\mathbf{x}))^2]. \quad (4)$$

Here  $\mathbf{v} \sim p_{\mathbf{v}}$  and  $\mathbf{x} \sim p_d$  are independent, and we require  $\mathbb{E}_{p_{\mathbf{v}}} [\mathbf{v} \mathbf{v}^T] \succ 0$  and  $\mathbb{E}_{p_{\mathbf{v}}} [\|\mathbf{v}\|_2^2] < \infty$ . Many examples of  $p_{\mathbf{v}}$  satisfy these requirements. For instance,  $p_{\mathbf{v}}$  can be a multivariate standard normal ( $\mathcal{N}(0, I_D)$ ), a multivariate Rademacher distribution (the uniform distribution over  $\{\pm 1\}^D$ ), or a uniform distribution on the hypersphere  $\mathbb{S}^{D-1}$  (recall that  $D$  refers to the dimension of  $\mathbf{x}$ ).

To eliminate the dependence of  $L(\theta; p_{\mathbf{v}})$  on  $s_d(\mathbf{x})$ , we use integration by parts as in score matching (cf., the equivalence between Eq. (1) and (2)). Defining

$$\begin{aligned}J(\theta; p_{\mathbf{v}}) &\triangleq \mathbb{E}_{p_{\mathbf{v}}} \mathbb{E}_{p_d} [\mathbf{v}^T \nabla_{\mathbf{x}} s_m(\mathbf{x}; \theta) \mathbf{v} \\ &\quad + \frac{1}{2} (\mathbf{v}^T s_m(\mathbf{x}; \theta))^2],\end{aligned} \quad (5)$$

the equivalence is summarized in the following theorem.

**Theorem 1.** *Under some regularity conditions (Assumption 1-3 in Appendix B.2), we have*

$$L(\theta; p_{\mathbf{v}}) = J(\theta; p_{\mathbf{v}}) + C, \quad (6)$$

where  $C$  is a constant w.r.t.  $\theta$ .

Other than our requirements on  $p_{\mathbf{v}}$ , the assumptions are exactly the same as in Theorem 1 of Hyvärinen (2005). We advise the interested readers to read Appendix B.2 for technical statements of the assumptions and a rigorous proof of the theorem.

In practice, given a dataset  $\mathbf{x}_1^N$ , we draw  $M$  projection vectors independently for each point  $\mathbf{x}_i$  from  $p_{\mathbf{v}}$ . The collection of all such vectors  $\{\mathbf{v}_{ij}\}_{1 \leq i \leq N, 1 \leq j \leq M}$  are abbreviated as  $\mathbf{v}_{11}^{NM}$ . We then use the following unbiased estimator of  $J(\theta; p_{\mathbf{v}})$

$$\begin{aligned}\hat{J}(\theta; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) &\triangleq \frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M \mathbf{v}_{ij}^T \nabla_{\mathbf{x}} s_m(\mathbf{x}_i; \theta) \mathbf{v}_{ij} \\ &\quad + \frac{1}{2} (\mathbf{v}_{ij}^T s_m(\mathbf{x}_i; \theta))^2.\end{aligned} \quad (7)$$

Note that when  $p_{\mathbf{v}}$  is a multivariate standard normal or multivariate Rademacher distribution, we have  $\mathbb{E}_{p_{\mathbf{v}}} [(\mathbf{v}^T s_m(\mathbf{x}; \theta))^2] = \|s_m(\mathbf{x}; \theta)\|_2^2$ , in which case the second term of  $J(\theta; p_{\mathbf{v}})$  can be integrated analytically, and may lead to an estimator with reduced variance:

$$\begin{aligned}\hat{J}_{\text{vr}}(\theta; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) &\triangleq \frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M \mathbf{v}_{ij}^T \nabla_{\mathbf{x}} s_m(\mathbf{x}_i; \theta) \mathbf{v}_{ij} \\ &\quad + \frac{1}{2} \|s_m(\mathbf{x}_i; \theta)\|_2^2.\end{aligned} \quad (8)$$

Empirically, we do find that  $\hat{J}_{\text{vr}}$  has better performance than  $\hat{J}$ . We refer to this version as sliced score matching with variance reduction (SSM-VR). In fact, we can

leverage  $\mathbb{E}_{p_{\mathbf{v}}}[(\mathbf{v}^\top \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))^2] = \|\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})\|_2^2$  to create a control variate for guaranteed reduction of variance (Appendix D).  $\hat{J}_{\text{vr}}$  is also closely related to Hutchinson's trace estimator (Hutchinson, 1990), which we will analyze later in Section 4.3.

For sliced score matching, the second derivative term  $\mathbf{v}^\top \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v}$  is much less computationally expensive than  $\text{tr}(\nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))$ . Using auto-differentiation systems that support higher order gradients, we can compute it using two gradient operations for a single  $\mathbf{v}$ , as shown in Alg. 1. Similarly, when there are  $M$   $\mathbf{v}$ 's, the total number of gradient operations is  $M + 1$ . In contrast, assuming the dimension of  $\mathbf{x}$  is  $D$  and  $D \gg M$ , we typically need  $D + 1$  gradient operations to compute  $\text{tr}(\nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))$  because each diagonal entry of  $\nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})$  needs to be computed separately (see Alg. 2 in the appendix).

---

**Algorithm 1** Sliced Score Matching

---

**Input:**  $\tilde{p}_m(\cdot; \boldsymbol{\theta}), \mathbf{x}, \mathbf{v}$

- 1:  $\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) \leftarrow \text{grad}(\log \tilde{p}_m(\mathbf{x}; \boldsymbol{\theta}), \mathbf{x})$
- 2:  $\mathbf{v}^\top \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) \leftarrow \text{grad}(\mathbf{v}^\top \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}), \mathbf{x})$
- 3:  $J \leftarrow \frac{1}{2}(\mathbf{v}^\top \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))^2$  (or  $J \leftarrow \frac{1}{2} \|\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})\|_2^2$ )
- 4:  $J \leftarrow J + \mathbf{v}^\top \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v}$

**return**  $J$

---

In practice, we can tune  $M$  to trade off variance and computational cost. In our experiments, we find that oftentimes  $M = 1$  is already a good choice.

### 3.2 SLICED SCORE ESTIMATION

As mentioned in section 2.2, the task of score estimation is to estimate  $\nabla_{\mathbf{x}} \log q(\mathbf{x})$  at some test point  $\mathbf{x}$ , given a set of samples  $\mathbf{x}_1^N \stackrel{\text{i.i.d.}}{\sim} q(\mathbf{x})$ . In what follows, we show how our sliced score matching objective  $\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM})$  can be straightforwardly adapted for this task.

We propose to use a vector-valued deep neural network  $\mathbf{h}(\mathbf{x}; \boldsymbol{\theta}) : \mathbb{R}^D \rightarrow \mathbb{R}^D$  as our score model. Then, substituting  $\mathbf{h}$  into  $\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM})$  for  $\mathbf{s}_m$ , we get

$$\frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M \mathbf{v}_{ij}^\top \nabla_{\mathbf{x}} \mathbf{h}(\mathbf{x}_i; \boldsymbol{\theta}) \mathbf{v}_{ij} + \frac{1}{2} (\mathbf{v}_{ij}^\top \mathbf{h}(\mathbf{x}_i; \boldsymbol{\theta}))^2,$$

and we can optimize the objective to get  $\hat{\boldsymbol{\theta}}$ . Afterwards,  $\mathbf{h}(\mathbf{x}; \hat{\boldsymbol{\theta}})$  can be used as an approximation to  $\nabla_{\mathbf{x}} \log q(\mathbf{x})$ .<sup>1</sup>

<sup>1</sup>Note that  $\mathbf{h}(\mathbf{x}; \boldsymbol{\theta})$  may not correspond to the gradient of any scalar function. For  $\mathbf{h}(\mathbf{x}; \boldsymbol{\theta})$  to represent a gradient, one necessary condition is  $\nabla_{\mathbf{x}} \times \mathbf{h}(\mathbf{x}; \boldsymbol{\theta}) = \mathbf{0}$  for all  $\mathbf{x}$ , which may not be satisfied by general networks. However, this is oblivious to the fact that  $\mathbf{h}(\mathbf{x}; \hat{\boldsymbol{\theta}})$  will be close to  $\nabla_{\mathbf{x}} \log q_{\boldsymbol{\theta}}(\mathbf{x})$  in  $\ell_2$  norm. As will be shown later, our argument based on integration by parts does not require  $\mathbf{h}(\mathbf{x}; \boldsymbol{\theta})$  to be a gradient.

Using a similar argument of integration by parts (cf., Eq. (4), (5) and (6)), we have

$$\begin{aligned} \mathbb{E}_{p_{\mathbf{v}}} \mathbb{E}_{p_d} \left[ \mathbf{v}^\top \nabla_{\mathbf{x}} \mathbf{h}(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v} + \frac{1}{2} (\mathbf{v}^\top \mathbf{h}(\mathbf{x}; \boldsymbol{\theta}))^2 \right] = \\ \frac{1}{2} \mathbb{E}_{p_{\mathbf{v}}} \mathbb{E}_{p_d} [(\mathbf{v}^\top \mathbf{h}(\mathbf{x}; \boldsymbol{\theta}) - \mathbf{v}^\top \nabla_{\mathbf{x}} \log q(\mathbf{x}))^2] + C, \end{aligned}$$

which implies that minimizing  $\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM})$  with  $\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})$  replaced by  $\mathbf{h}(\mathbf{x}; \boldsymbol{\theta})$  is approximately minimizing the average projected difference between  $\mathbf{h}(\mathbf{x}; \boldsymbol{\theta})$  and  $\nabla_{\mathbf{x}} \log q(\mathbf{x})$ . Hence,  $\mathbf{h}(\mathbf{x}; \hat{\boldsymbol{\theta}})$  should be close to  $\nabla_{\mathbf{x}} \log q(\mathbf{x})$  and can serve as a score estimator.

## 4 THEORETICAL ANALYSIS

In this section, we present several theoretical results to justify sliced score matching as a principled objective. We also discuss the connection of sliced score matching to other methods. For readability, we will defer rigorous statements of assumptions and theorems to the Appendix.

### 4.1 CONSISTENCY

One important question to ask for any statistical estimation objective is whether the estimated parameter is consistent under reasonable assumptions. Our results confirm that for any  $M$ , the objective  $\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM})$  is consistent under suitable assumptions as  $N \rightarrow \infty$ .

We need several standard assumptions to prove the results rigorously. Let  $p_m$  be the normalized density induced by our unnormalized model  $\tilde{p}_m$ , which is assumed to be normalizable. First, we assume  $\Theta$  is compact (Assumption 6), and our model family  $\{p_m(\mathbf{x}; \boldsymbol{\theta}) : \boldsymbol{\theta} \in \Theta\}$  is well-specified and identifiable (Assumption 4). These are standard assumptions used for proving the consistency of MLE (van der Vaart, 1998). We also adopt the assumption in Hyvärinen (2005) that all densities are strictly positive (Assumption 5). Finally, we assume that  $p_m(\mathbf{x}; \boldsymbol{\theta})$  has some Lipschitz properties (Assumption 7), and  $p_{\mathbf{v}}$  has bounded higher-order moments (Assumption 2, true for all  $p_{\mathbf{v}}$  considered in the experiments). Then, we can prove the consistency of  $\hat{\boldsymbol{\theta}}_{N,M} \triangleq \arg \min_{\boldsymbol{\theta} \in \Theta} \hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM})$ :

**Theorem 2** (Consistency). *Assume the conditions of Theorem 1 are satisfied. Assume further that the assumptions discussed above hold. Let  $\boldsymbol{\theta}^*$  be the true parameter of the data distribution. Then for every  $M \in \mathbb{N}^+$ ,*

$$\hat{\boldsymbol{\theta}}_{N,M} \xrightarrow{P} \boldsymbol{\theta}^*, \quad N \rightarrow \infty.$$

*Sketch of proof.* We first prove that  $J(\boldsymbol{\theta}; p_{\mathbf{v}}) = 0 \Leftrightarrow \boldsymbol{\theta} = \boldsymbol{\theta}^*$  (Lemma 1 and Theorem 1). Then we prove the uniform convergence of  $\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM})$  (Lemma 3), which

holds regardless of  $M$ . These two facts lead to consistency. For a complete proof, see Appendix B.3.  $\square$

**Remark 1.** In Hyvärinen (2005), the authors only showed that  $J(\boldsymbol{\theta}) = 0 \Leftrightarrow \boldsymbol{\theta} = \boldsymbol{\theta}^*$ , which leads to “local consistency” of score matching. This is a weaker notion of consistency compared to our settings.

## 4.2 ASYMPTOTIC NORMALITY

Asymptotic normality results can be very useful for approximate hypothesis testing and comparing different estimators. Below we show that  $\hat{\boldsymbol{\theta}}_{N,M}$  is asymptotically normal when  $N \rightarrow \infty$ .

In addition to the assumptions in Section 4.1, we need an extra assumption to prove asymptotic normality. We require  $p_m(\mathbf{x}; \boldsymbol{\theta})$  to have a stronger Lipschitz property (Assumption 9).

For simplicity, we denote  $\nabla_{\mathbf{x}} h(\mathbf{x})|_{\mathbf{x}=\mathbf{x}'}$  as  $\nabla_{\mathbf{x}} h(\mathbf{x}')$ , where  $h(\cdot)$  is an arbitrary function. In the following, we will only show the asymptotic normality result for a specific  $p_{\mathbf{v}}$ . More general results are in Appendix B.4.

**Theorem 3** (Asymptotic normality, special case). *Assume the assumptions discussed above hold. If  $p_{\mathbf{v}}$  is the multivariate Rademacher distribution, we have*

$$\sqrt{N}(\hat{\boldsymbol{\theta}}_{N,M} - \boldsymbol{\theta}^*) \xrightarrow{d} \mathcal{N}(0, \Sigma),$$

where

$$\Sigma \triangleq [\nabla_{\boldsymbol{\theta}}^2 J(\boldsymbol{\theta}^*)]^{-1} \left( \sum_{1 \leq i,j \leq D} V_{ij} + \frac{2}{M} \sum_{1 \leq i \neq j \leq D} W_{ij} \right) \cdot [\nabla_{\boldsymbol{\theta}}^2 J(\boldsymbol{\theta}^*)]^{-1}. \quad (9)$$

Here  $D$  is the dimension of data;  $V_{ij}$  and  $W_{ij}$  are p.s.d matrices depending on  $p_m(\mathbf{x}; \boldsymbol{\theta}^*)$ , and their definitions can be found in Appendix B.4.

*Sketch of proof.* Once we get the consistency (Theorem 2), the rest closely follows the proof of asymptotic normality of MLE (van der Vaart, 1998). A rigorous proof can be found in Appendix B.4.  $\square$

**Remark 2.** As expected, larger  $M$  will lead to smaller asymptotic variance, as can be seen in Eq. (9).

**Remark 3.** As far as we know, there is no published proof of asymptotic normality for the standard (not sliced) score matching. However, by using the same techniques in our proofs, and under similar assumptions, we can conclude that the asymptotic variance of the score matching estimator is  $[\nabla_{\boldsymbol{\theta}}^2 J(\boldsymbol{\theta}^*)]^{-1} \left( \sum_{ij} V_{ij} \right) [\nabla_{\boldsymbol{\theta}}^2 J(\boldsymbol{\theta}^*)]^{-1}$  (Corollary 1), which will always be smaller than sliced score matching with multivariate Rademacher projections. However, the gap reduces when  $M$  increases.

## 4.3 CONNECTION TO OTHER METHODS

Sliced score matching is widely connected to many other methods, and can be motivated from some different perspectives. Here we discuss a few of them.

**Connection to NCE.** Noise contrastive estimation (NCE), proposed by Gutmann & Hyvärinen (2010), is another principle for training unnormalized statistical models. The method works by comparing  $p_m(\mathbf{x}; \boldsymbol{\theta})$  with a noise distribution  $p_n(\mathbf{x})$ . We consider a special form of NCE which minimizes the following objective

$$-\mathbb{E}_{p_d}[\log h(\mathbf{x}; \boldsymbol{\theta})] - \mathbb{E}_{p_n}[\log(1 - h(\mathbf{x}; \boldsymbol{\theta}))], \quad (10)$$

where  $h(\mathbf{x}; \boldsymbol{\theta}) \triangleq \frac{p_m(\mathbf{x}; \boldsymbol{\theta})}{p_m(\mathbf{x}; \boldsymbol{\theta}) + p_m(\mathbf{x}-\mathbf{v}; \boldsymbol{\theta})}$ , and we choose  $p_n(\mathbf{x}) = p_d(\mathbf{x}+\mathbf{v})$ . Assuming  $\|\mathbf{v}\|_2$  to be small, Eq. (10) can be written as the following by Taylor expansion

$$\begin{aligned} \frac{1}{4} \mathbb{E}_{p_d} & \left[ \mathbf{v}^\top \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v} + \frac{1}{2} (\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})^\top \mathbf{v})^2 \right] \\ & + 2 \log 2 + o(\|\mathbf{v}\|_2^2). \end{aligned} \quad (11)$$

For derivation, see Proposition 1 in the appendix. A similar derivation can also be found in Gutmann & Hirayama (2011). As a result of (11), if we choose some  $p_{\mathbf{v}}$  and take the expectation of (10) w.r.t.  $p_{\mathbf{v}}$ , the objective will be equivalent to sliced score matching whenever  $\|\mathbf{v}\|_2 \approx 0$ .

**Connection to Hutchinson’s Trick.** Hutchinson’s trick (Hutchinson, 1990) is a stochastic algorithm to approximate  $\text{tr}(A)$  for any square matrix  $A$ . The idea is to choose a distribution of a random vector  $\mathbf{v}$  such that  $\mathbb{E}_{p_{\mathbf{v}}}[\mathbf{v}\mathbf{v}^\top] = I$ , and then we have  $\text{tr}(A) = \mathbb{E}_{p_{\mathbf{v}}}[\mathbf{v}^\top A \mathbf{v}]$ . Hence, using Hutchinson’s trick, we can replace  $\text{tr}(\nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))$  with  $\mathbb{E}_{p_{\mathbf{v}}}[\mathbf{v}^\top \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v}]$  in the score matching objective  $J(\boldsymbol{\theta})$ . Then the finite sample objective of score matching becomes

$$\frac{1}{N} \sum_{i=1}^N \left( \frac{1}{M} \sum_{j=1}^M \mathbf{v}_{ij}^\top \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}_i; \boldsymbol{\theta}) \mathbf{v}_{ij} \right) + \frac{1}{2} \|\mathbf{s}_m(\mathbf{x}_i; \boldsymbol{\theta})\|_2^2,$$

which is exactly the sliced score matching objective with variance reduction  $\hat{J}_{\text{vr}}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM})$ .

## 5 RELATED WORK

### 5.1 SCALABLE SCORE MATCHING

To the best of our knowledge, there are three existing methods that are able to scale up score matching to learning deep models on high dimensional data.

**Denoising Score Matching.** Vincent (2011) proposes denoising score matching, a variant of score matching that completely circumvents the Hessian. Specifically, consider a noise distribution  $q_\sigma(\tilde{\mathbf{x}} | \mathbf{x})$ , and let  $q_\sigma(\tilde{\mathbf{x}}) = \int q_\sigma(\tilde{\mathbf{x}} | \mathbf{x}) p_d(\mathbf{x}) d\mathbf{x}$ . Denoising score matching applies the original score matching to the noise-corrupted data distribution  $q_\sigma(\tilde{\mathbf{x}})$ , and the objective can be proven to be equivalent to the following up to a constant

$$\frac{1}{2} \mathbb{E}_{q_\sigma(\tilde{\mathbf{x}} | \mathbf{x}) p_d(\mathbf{x})} [\|\mathbf{s}_m(\tilde{\mathbf{x}}; \boldsymbol{\theta}) - \nabla_{\mathbf{x}} \log q_\sigma(\tilde{\mathbf{x}} | \mathbf{x})\|_2^2],$$

which can be estimated without computing any Hessian. Although denoising score matching is much faster than score matching, it has obvious drawbacks. First, it can only recover the noise corrupted data distribution. Furthermore, choosing the parameters of the noise distribution is highly non-trivial and in practice the performance can be very sensitive to  $\sigma$ , and heuristics have to be used in practice. For example, Saremi et al. (2018) propose a heuristic for choosing  $\sigma$  based on Parzen windows.

**Approximate Backpropagation.** Kingma & LeCun (2010) propose a backpropagation method to approximately compute the trace of the Hessian. Because the backpropagation of the full Hessian scales quadratically w.r.t. the layer size, the authors limit backpropagation only to the diagonal so that it has the same cost as the usual backpropagation for computing loss gradients. However, there are no theoretical guarantees for the approximation errors. In fact, the authors only did experiments on networks with a single hidden layer, in which case the approximation is exact. Moreover, there is no direct support for the proposed approximate backpropagation method in modern automatic differentiation frameworks.

**Curvature Propagation.** Martens et al. (2012) estimate the trace term in score matching by applying curvature propagation to compute an unbiased, complex-valued estimator of the diagonal of the Hessian. The work claims that curvature propagation will have the smallest variance among a class of estimators, which includes the Hutchinson’s estimator. However, their proof evaluates the pseudo-variance of the complex-valued estimator instead of the variance. In practice, curvature propagation can have large variance when the number of nodes in the network is large, because it introduces noise for each node in the network. Finally, implementing curvature propagation also requires manually modifying the backpropagation code, handling complex numbers in neural networks, and will be inefficient for networks of more general structures, such as recurrent neural networks.

## 5.2 KERNEL SCORE ESTIMATORS

Two prior methods for score estimation are based on a generalized form of Stein’s identity (Stein, 1981; Gorham & Mackey, 2017):

$$\mathbb{E}_{q(\mathbf{x})} [\mathbf{h}(\mathbf{x}) \nabla_{\mathbf{x}} \log q(\mathbf{x})^\top + \nabla_{\mathbf{x}} \mathbf{h}(\mathbf{x})] = \mathbf{0}, \quad (12)$$

where  $q(\mathbf{x})$  is a continuously differentiable density and  $\mathbf{h}(\mathbf{x})$  is a function satisfying some regularity conditions. Li & Turner (2018) propose to set  $\mathbf{h}(\mathbf{x})$  as the feature map of some kernel in the *Stein class* (Liu et al., 2016) of  $q$ , and solve a finite-sample version of (12) to obtain estimates of  $\nabla_{\mathbf{x}} \log q(\mathbf{x})$  at the sample points. We refer to this method as *Stein* in the experiments. Shi et al. (2018) adopt a different approach but also exploit (12). They build their estimator by expanding  $\nabla_{\mathbf{x}} \log q(\mathbf{x})$  as a spectral series of eigenfunctions and solve for the coefficients using (12). Compared to *Stein*, their method is argued to have theoretical guarantees and principled out-of-sample estimation at an unseen test point. We refer to their method as *Spectral* in the experiments.

## 6 EXPERIMENTS

Our experiments include two parts: (1) to test the effectiveness of sliced score matching (**SSM**) in learning deep models for density estimation, and (2) to test the ability of **SSM** in providing score estimates for applications such as VAEs with implicit encoders and WAEs. Unless specified explicitly, we choose  $M = 1$  by default.

### 6.1 DENSITY ESTIMATION

We evaluate **SSM** and its variance-reduced version (**SSM-VR**) for density estimation and compare against score matching (**SM**) and its three scalable baselines: denoising score matching (**DSM**), approximate backpropagation (**approx BP**), and curvature propagation (**CP**). All **SSM** methods in this section use the multivariate Rademacher distribution as  $p_v$ . Our results demonstrate that: (1) **SSM** is comparable in performance to **SM**, (2) **SSM** outperforms other computationally efficient approximations to **SM**, and (3) unlike **SM**, **SSM** scales effectively to high dimensional data.

#### 6.1.1 Deep Kernel Exponential Families

**Model.** Deep kernel exponential families (DKEF) are unnormalized density models trained using **SM** (Wenliang et al., 2019). DKEFs parameterize the unnormalized log density as  $\log \tilde{p}_f(\mathbf{x}) = f(\mathbf{x}) + \log q_0(\mathbf{x})$ , where  $f$  is a mixture of Gaussian kernels evaluated at different inducing points:  $f(\mathbf{x}) = \sum_{l=1}^L \alpha_l k(\mathbf{x}, \mathbf{z}_l)$ . The kernel is defined on the feature space of a neural network, and the

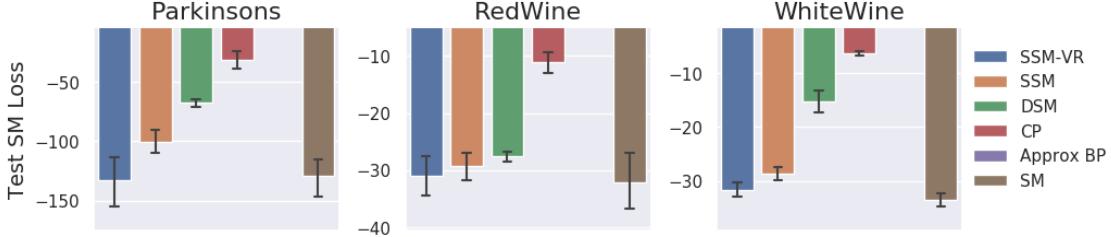


Figure 1: **SM** loss after training DKEF models on UCI datasets with different loss functions; lower is better. Results for approximate backpropagation are not shown because losses were larger than  $10^9$ .

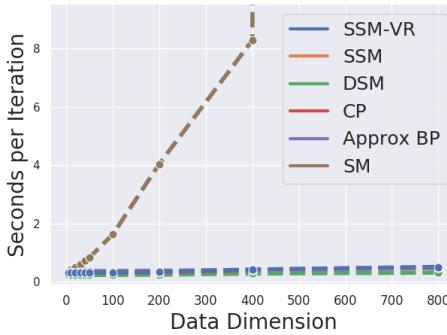


Figure 2: **SM** performance degrades linearly with the data dimension, while efficient approaches have relatively similar performance.

network parameters are trained along with the inducing points  $\mathbf{z}_l$ . Further details of the model, which is identical to that in Wenliang et al. (2019), are presented in Appendix C.1.

**Setup.** Following the settings of Wenliang et al. (2019), we trained models on three UCI datasets: Parkinsons, RedWine, and WhiteWine (Dua & Graff, 2017), and used their original code for **SM**. To compute the trace term exactly, Wenliang et al. (2019)’s manual implementation of backpropagation takes over one thousand lines for a model that is four layers deep, while the implementation of **SSM** only takes several lines. For **DSM**, the value of  $\sigma$  is chosen by grid search using the **SM** loss on a validation dataset. All models are trained with 15 different random seeds and training is stopped when validation loss does not improve for 200 steps.

**Results.** Results in Fig. 1 demonstrate that **SSM-VR** performs comparably to **SM**, when evaluated using the **SM** loss on a held-out test set. We observe that variance reduction substantially improves the performance of **SSM**. In addition, **SSM** outperforms other computationally efficient approaches. **DSM** can perform comparably to **SSM** on RedWine. However, it is challenging to select  $\sigma$  for **DSM**. Models trained using  $\sigma$  from the heuristic in Saremi

	Test <b>SM</b> Loss	Test LL
MLE	-579	<b>-791</b>
<b>SSM-VR</b>	<b>-8054</b>	-3355
<b>SSM</b>	-2428	-2039
<b>DSM</b> ( $\sigma = 0.10$ )	-3035	-4363
<b>DSM</b> ( $\sigma = 1.74$ )	-97	-8082
<b>CP</b>	-1694	-1517
<b>Approx BP</b>	-48	-2288

Table 1: Score matching losses and log-likelihoods for NICE models on MNIST.  $\sigma = 0.1$  is by grid search and  $\sigma = 1.74$  is from the heuristic of Saremi et al. (2018).

et al. (2018) are far from optimal (on both **SM** losses and likelihoods), and extensive grid search is needed to find the best  $\sigma$ . **CP** performs substantially worse, which is likely because it injects noise for each node in the computation graph, and the amount of noise introduced is too large for a neural-network-based kernel evaluated at 200 inducing points, which supports our hypothesis that **CP** does not work effectively for deep models. Results for **approx BP** are omitted because the losses exceeded  $10^9$  on all datasets. This is because **approx BP** provides a biased estimate of the Hessian without any error guarantees. All the results are similar when evaluating according to log-likelihood metric (Appendix C.1).

**Scalability to High Dimensional Data.** We evaluate the computational efficiency of different losses on data of increasing dimensionality. We fit DKEF models to a multivariate standard normal in high dimensional spaces. The average running time per minibatch of 100 examples is reported in Fig. 2. **SM** performance degrades linearly with the dimension of the input data due to the computation of the Hessian, and creates out of memory errors for a 12GB GPU after the dimension increases beyond 400. **SSM** performs similarly to **DSM**, **approx BP** and **CP**, and they are all scalable to large dimensions.

### 6.1.2 Deep Flow Models

**Setup.** As a sanity check, we also evaluate **SSM** by training a NICE flow model (Dinh et al., 2015), whose likelihood is tractable and can be compared to results obtained by **MLE**. The model we use has 20 hidden layers, and 1000 units per layer. Models are trained to fit MNIST handwritten digits, which are 784 dimensional images. Data are dequantized by adding uniform noise in the range  $[-\frac{1}{512}, \frac{1}{512}]$ , and transformed using a logit transformation,  $\log(x) - \log(1 - x)$ . We provide additional details in Appendix C.2.

Training with **SM** is extremely computationally expensive in this case. Our **SM** implementation based on auto-differentiation takes around 7 hours to finish one epoch of training, and 12 GB GPU memory cannot hold a batch size larger than 24, so we do not include these results. Since NICE has tractable likelihoods, we also evaluate **MLE** as a surrogate objective for minimizing the **SM** loss. Notably, likelihoods and **SM** losses might be uncorrelated when the model is mis-specified, which is likely to be the case for a complex dataset like MNIST.

**Results.** **SM** losses and log-likelihoods on the test dataset are reported in Tab. 1, where models are evaluated using the best checkpoint in terms of the **SM** loss on a validation dataset over 100 epochs of training. **SSM-VR** greatly outperforms all the other methods on the **SM** loss. **DSM** performs worse than **SSM-VR**, and  $\sigma$  is still hard to tune. Specifically, following the heuristic in Saremi et al. (2018) leads to  $\sigma = 1.74$ , which performed the worst (on both log-likelihood and **SM** loss) of the eight choices of  $\sigma$  in our grid search. **Approx BP** has more success on NICE than for training DKEF models. This might be because the Hessians of hidden layers of NICE are closer to a diagonal matrix, which results in a smaller approximation error for **approx BP**. As in the DKEF experiments, **CP** performs worse. This is likely due to injecting noise to all hidden units, which will lead to large variance for a network as big as NICE. Unlike the DKEF experiments, we find that good log-likelihoods are less correlated with good **SM** loss, probably due to model mis-specification.

## 6.2 SCORE ESTIMATION

We consider two typical tasks that require accurate score estimations: (1) training VAEs with an implicit encoder and (2) training Wasserstein Auto-Encoders. We show in both tasks **SSM** outperforms previous score estimators. Samples generated by various algorithms can be found in Appendix A.

Latent Dim	VAE		WAE	
	8	32	8	32
ELBO	96.87	<b>89.06</b>	N/A	N/A
<b>SSM</b>	<b>95.50</b>	89.25 ( <b>88.29</b> <sup>†</sup> )	<b>98.24</b>	<b>90.37</b>
Stein	96.71	91.84	99.05	91.70
Spectral	96.60	94.67	98.81	92.55

Table 2: Negative log-likelihoods on MNIST, estimated by AIS. <sup>†</sup>The result of **SSM** with  $M = 100$ , in which case **SSM** matches the computational cost of kernel methods, which used 100 samples for each data point.

	Method \ Iteration	10k	40k	70k	100k
		ELBO	VAE	WAE	VAE
VAE	<b>SSM</b>	108.52	<b>70.28</b>	<b>66.52</b>	<b>62.50</b>
	Stein	126.60	118.87	120.51	126.76
	Spectral	131.90	125.04	128.36	133.93
	<b>SSM</b>	84.11	<b>61.09</b>	<b>56.23</b>	<b>54.33</b>
WAE	Stein	82.93	63.46	58.53	57.61
	Spectral	<b>82.30</b>	62.47	58.03	55.96

Table 3: FID scores of different methods versus number of training iterations on CelebA dataset.

### 6.2.1 VAE with Implicit Encoders

**Background.** Consider a latent variable model  $p(\mathbf{x}, \mathbf{z})$ , where  $\mathbf{x}$  and  $\mathbf{z}$  denote observed and latent variables respectively. A variational auto-encoder (VAE) is composed of two parts: 1) an encoder  $p_\theta(\mathbf{x} | \mathbf{z})$  modeling the conditional distribution of  $\mathbf{x}$  given  $\mathbf{z}$ ; and a decoder  $q_\phi(\mathbf{z} | \mathbf{x})$  that approximates the posterior distribution of the latent variable. A VAE is typically trained by maximizing the following evidence lower bound (ELBO)

$$\mathbb{E}_{p_d} [\mathbb{E}_{q_\phi(\mathbf{z} | \mathbf{x})} \log p_\theta(\mathbf{x} | \mathbf{z}) p(\mathbf{z}) - \mathbb{E}_{q_\phi(\mathbf{z} | \mathbf{x})} \log q_\phi(\mathbf{z} | \mathbf{x})],$$

where  $p(\mathbf{z})$  denotes a pre-specified prior distribution. The expressive power of  $q_\phi(\mathbf{z} | \mathbf{x})$  is critical to the success of variational learning. Typically,  $q_\phi(\mathbf{z} | \mathbf{x})$  is chosen to be a simple distribution with tractable densities so that  $H(q_\phi) \triangleq -\mathbb{E}_{q_\phi(\mathbf{z} | \mathbf{x})} \log q_\phi(\mathbf{z} | \mathbf{x})$  is tractable. We call this traditional approach “ELBO” in the experiments. With score estimation techniques, we can directly compute  $\nabla_\phi H(q_\phi)$  for implicit distributions, which enables more flexible options for  $q_\phi$ . We consider 3 score estimation techniques: **SSM**, **Stein** (Li & Turner, 2018) and **Spectral** (Shi et al., 2018).

For a single data point  $\mathbf{x}$ , kernel score estimators need multiple samples from  $q_\phi(\mathbf{z} | \mathbf{x})$  to estimate its score. On MNIST, we use 100 samples, as done in Shi et al. (2018). On CelebA, however, we can only take 20 samples due to GPU memory limitations. In contrast, **SSM** learns a score

network  $h(\mathbf{z} \mid \mathbf{x})$  along with  $q_\phi(\mathbf{z} \mid \mathbf{x})$ , which amortizes the cost of score estimation. Unless noted explicitly, we use one projection per data point ( $M = 1$ ) for **SSM**, which is scalable to deep networks.

**Setup.** We consider VAE training on MNIST and CelebA (Liu et al., 2015). All images in CelebA are first cropped to a patch of  $140 \times 140$  and then resized to  $64 \times 64$ . We report negative likelihoods on MNIST, as estimated by AIS (Neal, 2001) with 1000 intermediate distributions. We evaluate sample quality on CelebA with FID scores (Heusel et al., 2017). For fast AIS evaluation on MNIST, we use shallow fully-connected networks with 3 hidden layers. For CelebA experiments we use deep convolutional networks. The architectures of implicit encoders and score networks are straightforward modifications to the encoders of **ELBO**. More details are provided in Appendix C.3.

**Results.** The negative likelihoods of different methods on MNIST are reported in the left part of Tab. 2. We note that **SSM** outperforms **Stein** and **Spectral** in all cases. When the latent dimension is 8, **SSM** outperforms **ELBO**, which indicates that the expressive power of implicit  $q_\phi(\mathbf{z} \mid \mathbf{x})$  pays off. When the latent dimension is 32, the gaps between **SSM** and kernel methods are even larger, and the performance of **SSM** is still comparable to **ELBO**. Moreover, when  $M = 100$  (matching the computation of kernel methods), **SSM** outperforms **ELBO**.

For CelebA, we provide FID scores of samples in the top part of Tab. 3. We observe that after 40k training iterations, **SSM** outperforms all other baselines. Kernel methods perform poorly in this case because only 20 samples per data point can be used due to limited amount of GPU memory. Early during training, **SSM** does not perform as well. Since the score network is trained along with the encoder and decoder, a moderate number of training steps is needed to give an accurate score estimation (and better learning of the VAE).

### 6.2.2 WAEs

**Background.** WAE is another method to learn latent variable models, which generally produces better samples than VAEs. Similar to a VAE, it contains an encoder  $q_\phi(\mathbf{z} \mid \mathbf{x})$  and a decoder  $p_\theta(\mathbf{x} \mid \mathbf{z})$  and both can be implicit distributions. Let  $p(\mathbf{z})$  be a pre-defined prior distribution, and  $q_\phi(\mathbf{z}) \triangleq \int q_\phi(\mathbf{z} \mid \mathbf{x}) p_d(\mathbf{x}) d\mathbf{x}$  denote the aggregated posterior distribution. Using a metric function  $c(\cdot, \cdot)$  and KL divergence between  $q_\phi(\mathbf{z})$  and  $p(\mathbf{z})$ , WAE minimizes the following objective

$$\mathbb{E}_{p_d} [\mathbb{E}_{q_\phi(\mathbf{z} \mid \mathbf{x})} [c(\mathbf{x}, p_\theta(\mathbf{x} \mid \mathbf{z})) - \lambda \log p(\mathbf{z})]] - \lambda H(q_\phi(\mathbf{z})).$$

Compared to  $\nabla_\phi H(q_\phi(\mathbf{z} \mid \mathbf{x}))$ , it is easier to estimate  $\nabla_\phi H(q_\phi(\mathbf{z}))$  for kernel methods, because the samples of  $q_\phi(\mathbf{z})$  can be collected by first sampling  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N \stackrel{\text{i.i.d.}}{\sim} p_d(\mathbf{x})$  and then sample one  $\mathbf{z}$  for each  $\mathbf{x}_i$  from  $q_\phi(\mathbf{z} \mid \mathbf{x}_i)$ . In contrast, multiple  $\mathbf{z}$ 's need to be sampled for each  $\mathbf{x}_i$  when estimating  $\nabla_\phi H(q_\phi(\mathbf{z} \mid \mathbf{x}))$  with kernel approaches. For **SSM**, we use a score network  $h(\mathbf{z})$  and train it alongside  $q_\phi(\mathbf{z} \mid \mathbf{x})$ .

**Setup.** The setup for WAE experiments is largely the same as VAE. The architectures are very similar to those used in the VAE experiments, and the only difference is that we made decoders implicit, as suggested in Tolstikhin et al. (2018). More details can be found in Appendix C.3.

**Results.** The negative likelihoods on MNIST are provided in the right part of Tab. 2. **SSM** outperforms both kernel methods, and achieves a larger performance gap when the latent dimension is higher. The likelihoods are lower than the VAE results as the WAE objective does not directly maximize likelihoods.

We show FID scores for CelebA experiments in the bottom part of Tab. 3. As expected, kernel methods perform much better than before, because it is faster to sample from  $q_\phi(\mathbf{z})$ . The FID scores are generally lower than those in VAE experiments, which supports previous results that WAEs generally obtain better samples. **SSM** outperforms both kernel methods when the number of iterations is more than 40k. This shows the advantages of training a deep, expressive score network compared to using a fixed kernel in score estimation tasks.

## 7 CONCLUSION

We propose sliced score matching, a scalable method for learning unnormalized statistical models and estimating scores for implicit distributions. Compared to the original score matching and its variants, our estimator can scale to deep models on high dimensional data, while remaining easy to implement in modern automatic differentiation frameworks. Theoretically, our estimator is consistent and asymptotically normal under some regularity conditions. Experimental results demonstrate that our method outperforms competitors in learning deep energy-based models and provides more accurate estimates than kernel score estimators in training implicit VAEs and WAEs.

### Acknowledgements

This research was supported by Intel Corporation, TRI, NSF (#1651565, #1522054, #1733686), ONR (N00014-19-1-2145), AFOSR (FA9550-19-1-0024).

## References

- Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pp. 265–283, 2016.
- Adam, P., Soumith, C., Gregory, C., Edward, Y., Zachary, D., Zeming, L., Alban, D., Luca, A., and Adam, L. Automatic differentiation in pytorch. In *NIPS Autodiff Workshop*, 2017.
- Canu, S. and Smola, A. Kernel methods and the exponential family. *Neurocomputing*, 69(7):714 – 720, 2006. ISSN 0925-2312. New Issues in Neurocomputing: 13th European Symposium on Artificial Neural Networks.
- Dinh, L., Krueger, D., and Bengio, Y. NICE: Non-linear independent components estimation. *International Conference in Learning Representations Workshop Track*, 2015.
- Dua, D. and Graff, C. UCI machine learning repository, 2017. URL <http://archive.ics.uci.edu/ml>.
- Dudley, R. M. The sizes of compact subsets of Hilbert space and continuity of Gaussian processes. *Journal of Functional Analysis*, 1(3):290–330, 1967.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.
- Gorham, J. and Mackey, L. Measuring sample quality with kernels. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 1292–1301, 2017.
- Gutmann, M. and Hyvärinen, A. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 297–304, 2010.
- Gutmann, M. U. and Hirayama, J.-i. Bregman divergence as general framework to estimate unnormalized statistical models. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pp. 283–290. AUAI Press, 2011.
- Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, pp. 6626–6637, 2017.
- Huszár, F. Variational inference using implicit distributions. *arXiv preprint arXiv:1702.08235*, 2017.
- Hutchinson, M. F. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 19(2):433–450, 1990.
- Hyvärinen, A. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(Apr):695–709, 2005.
- Kingma, D. P. and LeCun, Y. Regularized estimation of image statistics by score matching. In *Advances in Neural Information Processing Systems*, pp. 1126–1134, 2010.
- Kingma, D. P. and Welling, M. Auto-encoding variational Bayes. *International Conference on Learning Representations*, 2014.
- LeCun, Y., Chopra, S., Hadsell, R., Ranzato, M. A., and Huang, F. J. A tutorial on energy-based learning. In *Predicting structured data*. MIT Press, 2006.
- Li, Y. and Turner, R. E. Gradient estimators for implicit models. In *International Conference on Learning Representations*, 2018.
- Liu, Q., Lee, J., and Jordan, M. A kernelized Stein discrepancy for goodness-of-fit tests. In *Proceedings of The 33rd International Conference on Machine Learning*, pp. 276–284, 2016.
- Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.
- Martens, J., Sutskever, I., and Swersky, K. Estimating the Hessian by back-propagating curvature. *Proceedings of the 29th International Conference on Machine Learning*, 2012.
- Neal, R. M. Annealed importance sampling. *Statistics and computing*, 11(2):125–139, 2001.
- Owen, A. B. *Monte Carlo theory, methods and examples*. 2013.
- Rabin, J., Peyré, G., Delon, J., and Bernot, M. Wasserstein barycenter and its application to texture mixing. In Bruckstein, A. M., ter Haar Romeny, B. M., Bronstein, A. M., and Bronstein, M. M. (eds.), *Scale Space and Variational Methods in Computer Vision*, pp. 435–446, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-24785-9.
- Saremi, S., Mehrjou, A., Schölkopf, B., and Hyvärinen, A. Deep energy estimator networks. *arXiv preprint arXiv:1805.08306*, 2018.
- Sasaki, H., Hyvärinen, A., and Sugiyama, M. Clustering via mode seeking by direct estimation of the gradient of a log-density. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 19–34. Springer, 2014.

- Shi, J., Sun, S., and Zhu, J. A spectral approach to gradient estimation for implicit distributions. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 4651–4660, 2018.
- Sriperumbudur, B., Fukumizu, K., Gretton, A., Hyvärinen, A., and Kumar, R. Density estimation in infinite dimensional exponential families. *Journal of Machine Learning Research*, 18(57):1–59, 2017.
- Stein, C. M. Estimation of the mean of a multivariate normal distribution. *The annals of Statistics*, pp. 1135–1151, 1981.
- Strathmann, H., Sejdinovic, D., Livingstone, S., Szabo, Z., and Gretton, A. Gradient-free Hamiltonian monte carlo with efficient kernel exponential families. In *Advances in Neural Information Processing Systems*, pp. 955–963, 2015.
- Sun, S., Zhang, G., Shi, J., and Grosse, R. Functional variational Bayesian neural networks. In *International Conference on Learning Representations*, 2019.
- Sutherland, D., Strathmann, H., Arbel, M., and Gretton, A. Efficient and principled score estimation with Nyström kernel exponential families. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, pp. 652–660, 2018.
- Tolstikhin, I., Bousquet, O., Gelly, S., and Schoelkopf, B. Wasserstein auto-encoders. In *International Conference on Learning Representations*, 2018.
- Tran, D., Ranganath, R., and Blei, D. Hierarchical implicit models and likelihood-free variational inference. In *Advances in Neural Information Processing Systems*, pp. 5523–5533, 2017.
- van der Vaart, A. W. *Asymptotic Statistics*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1998. doi: 10.1017/CBO9780511802256.
- Vincent, P. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7): 1661–1674, 2011.
- Wenliang, L., Sutherland, D., Strathmann, H., and Gretton, A. Learning deep kernels for exponential family densities. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 6737–6746, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL <http://proceedings.mlr.press/v97/wenliang19a.html>.

## A Samples

### A.1 VAE WITH IMPLICIT ENCODERS

#### A.1.1 MNIST

	Latent Dim 8	Latent Dim 32
ELBO	0 9 2 7 4 0 7 3 9 0 0 7 5 4 5 6 9 8 9 4 4 6 4 9 9 2 2 8 6 0 7 3 3 7 0 0 9 0 3 8 1 8 8 3 4 4 6 9 8 6 8 4 8 8 8 1 8 7 5 9 5 6 0 9 0 8 1 1 4 3 6 8 0 9 0 1 7 0 7 7 9 8 2 9 9 7 9 3 6 4 9 8 9 2 0 8 7 7 5 4	4 5 9 6 0 1 8 4 8 2 2 1 8 0 8 3 1 3 9 7 4 0 9 8 9 3 8 4 0 7 9 9 5 3 7 5 0 0 1 7 6 2 9 7 5 5 2 4 9 9 3 8 2 4 3 4 9 1 8 0 3 9 8 9 8 3 6 7 2 0 0 3 0 3 4 9 8 3 3 3 6 4 1 2 8 3 4 6 4 8 8 2 2 8 4 7 9 6 4 9
	2 0 9 0 5 6 0 4 0 9 9 5 2 8 3 0 8 1 3 8 3 6 5 1 1 8 8 8 4 2 8 7 4 7 0 4 0 4 0 2 8 3 2 4 6 8 8 0 8 2 2 1 3 3 4 9 9 8 0 0 0 2 5 8 1 8 1 2 3 1 3 4 3 9 3 0 9 1 9 6 0 9 5 8 4 0 3 3 0 0 0 3 5 1 8 2 8 8 3 0 4 1	7 4 3 0 5 9 5 2 2 3 5 1 2 9 8 7 7 9 7 4 9 6 1 0 3 9 5 1 6 8 1 8 3 8 7 8 0 9 1 3 8 9 4 5 9 4 9 4 6 8 5 6 0 5 4 3 2 3 2 7 9 9 6 1 2 5 4 4 6 3 0 8 8 3 1 8 6 9 3 5 6 2 4 8 8 1 3 0 3 2 8 9 3 5 1 1 0 5 5 4
	5 8 9 1 1 3 0 1 9 9 0 4 2 6 2 6 1 1 9 8 9 7 0 0 0 9 7 1 0 0 4 8 2 7 2 8 2 8 8 8 9 3 2 4 4 4 0 6 1 0 8 4 9 3 8 1 8 4 2 8 8 7 0 9 9 3 9 9 0 3 5 8 8 4 9 1 1 2 5 4 6 8 0 8 8 0 4 0 6 9 9 8 6 0 6 1 0 0 3 5	2 2 2 5 4 5 1 2 2 3 1 2 8 6 0 3 0 0 3 2 5 3 1 0 6 3 4 2 2 2 3 7 3 2 0 5 4 0 8 4 0 4 5 9 3 5 1 0 9 2 3 3 1 6 2 5 8 4 3 1 5 4 6 0 7 2 5 1 1 2 1 5 3 1 3 6 2 5 8 1 8 5 4 5 7 4 9 7 0 2 7 3 0 9 6 1 8 3 3 5 6 3
	6 4 0 9 1 6 2 3 4 9 0 4 1 2 2 6 8 8 8 9 4 2 6 4 9 0 0 3 9 6 0 8 3 9 3 5 3 8 2 8 0 8 3 7 9 2 9 4 2 8 8 2 9 1 8 9 8 3 2 6 2 0 9 3 9 2 2 9 0 3 6 8 2 9 0 9 8 2 6 3 4 8 7 6 1 2 9 0 6 0 0 9 2 0 1 3 4 0 8 9	2 3 8 1 2 1 6 4 1 7 4 0 5 5 2 6 4 5 3 3 1 1 2 3 3 5 2 3 0 2 6 5 6 0 3 8 4 3 6 3 9 7 9 5 6 1 8 1 2 3 4 0 4 8 0 2 3 1 5 4 2 9 8 4 6 8 0 3 3 3 5 2 8 2 5 2 7 1 5 2 0 7 8 0 1 2 4 5 3 3 4 0 6 9 2 3 3 1 3
	2 0 9 1 1 3 0 1 9 9 0 4 2 6 2 6 1 1 9 8 9 7 0 0 0 9 7 1 0 0 4 8 2 7 2 8 2 8 8 8 9 3 2 4 4 4 0 6 1 0 8 4 9 3 8 1 8 4 2 8 8 7 0 9 9 3 9 9 0 3 5 8 8 4 9 1 1 2 5 4 6 8 0 8 8 0 4 0 6 9 9 8 6 0 6 1 0 0 3 5	2 2 2 5 4 5 1 2 2 3 1 2 8 6 0 3 0 0 3 2 5 3 1 0 6 3 4 2 2 2 3 7 3 2 0 5 4 0 8 4 0 4 5 9 3 5 1 0 9 2 3 3 1 6 2 5 8 4 3 1 5 4 6 0 7 2 5 1 1 2 1 5 3 1 3 6 2 5 8 1 8 5 4 5 7 4 9 7 0 2 7 3 0 9 6 1 8 3 3 5 6 3
	2 0 9 1 1 3 0 1 9 9 0 4 2 6 2 6 1 1 9 8 9 7 0 0 0 9 7 1 0 0 4 8 2 7 2 8 2 8 8 8 9 3 2 4 4 4 0 6 1 0 8 4 9 3 8 1 8 4 2 8 8 7 0 9 9 3 9 9 0 3 5 8 8 4 9 1 1 2 5 4 6 8 0 8 8 0 4 0 6 9 9 8 6 0 6 1 0 0 3 5	2 3 8 1 2 1 6 4 1 7 4 0 5 5 2 6 4 5 3 3 1 1 2 3 3 5 2 3 0 2 6 5 6 0 3 8 4 3 6 3 9 7 9 5 6 1 8 1 2 3 4 0 4 8 0 2 3 1 5 4 2 9 8 4 6 8 0 3 3 3 5 2 8 2 5 2 7 1 5 2 0 7 8 0 1 2 4 5 3 3 4 0 6 9 2 3 3 1 3
	2 0 9 1 1 3 0 1 9 9 0 4 2 6 2 6 1 1 9 8 9 7 0 0 0 9 7 1 0 0 4 8 2 7 2 8 2 8 8 8 9 3 2 4 4 4 0 6 1 0 8 4 9 3 8 1 8 4 2 8 8 7 0 9 9 3 9 9 0 3 5 8 8 4 9 1 1 2 5 4 6 8 0 8 8 0 4 0 6 9 9 8 6 0 6 1 0 0 3 5	2 3 8 1 2 1 6 4 1 7 4 0 5 5 2 6 4 5 3 3 1 1 2 3 3 5 2 3 0 2 6 5 6 0 3 8 4 3 6 3 9 7 9 5 6 1 8 1 2 3 4 0 4 8 0 2 3 1 5 4 2 9 8 4 6 8 0 3 3 3 5 2 8 2 5 2 7 1 5 2 0 7 8 0 1 2 4 5 3 3 4 0 6 9 2 3 3 1 3
	2 0 9 1 1 3 0 1 9 9 0 4 2 6 2 6 1 1 9 8 9 7 0 0 0 9 7 1 0 0 4 8 2 7 2 8 2 8 8 8 9 3 2 4 4 4 0 6 1 0 8 4 9 3 8 1 8 4 2 8 8 7 0 9 9 3 9 9 0 3 5 8 8 4 9 1 1 2 5 4 6 8 0 8 8 0 4 0 6 9 9 8 6 0 6 1 0 0 3 5	2 3 8 1 2 1 6 4 1 7 4 0 5 5 2 6 4 5 3 3 1 1 2 3 3 5 2 3 0 2 6 5 6 0 3 8 4 3 6 3 9 7 9 5 6 1 8 1 2 3 4 0 4 8 0 2 3 1 5 4 2 9 8 4 6 8 0 3 3 3 5 2 8 2 5 2 7 1 5 2 0 7 8 0 1 2 4 5 3 3 4 0 6 9 2 3 3 1 3
	2 0 9 1 1 3 0 1 9 9 0 4 2 6 2 6 1 1 9 8 9 7 0 0 0 9 7 1 0 0 4 8 2 7 2 8 2 8 8 8 9 3 2 4 4 4 0 6 1 0 8 4 9 3 8 1 8 4 2 8 8 7 0 9 9 3 9 9 0 3 5 8 8 4 9 1 1 2 5 4 6 8 0 8 8 0 4 0 6 9 9 8 6 0 6 1 0 0 3 5	2 3 8 1 2 1 6 4 1 7 4 0 5 5 2 6 4 5 3 3 1 1 2 3 3 5 2 3 0 2 6 5 6 0 3 8 4 3 6 3 9 7 9 5 6 1 8 1 2 3 4 0 4 8 0 2 3 1 5 4 2 9 8 4 6 8 0 3 3 3 5 2 8 2 5 2 7 1 5 2 0 7 8 0 1 2 4 5 3 3 4 0 6 9 2 3 3 1 3
	2 0 9 1 1 3 0 1 9 9 0 4 2 6 2 6 1 1 9 8 9 7 0 0 0 9 7 1 0 0 4 8 2 7 2 8 2 8 8 8 9 3 2 4 4 4 0 6 1 0 8 4 9 3 8 1 8 4 2 8 8 7 0 9 9 3 9 9 0 3 5 8 8 4 9 1 1 2 5 4 6 8 0 8 8 0 4 0 6 9 9 8 6 0 6 1 0 0 3 5	2 3 8 1 2 1 6 4 1 7 4 0 5 5 2 6 4 5 3 3 1 1 2 3 3 5 2 3 0 2 6 5 6 0 3 8 4 3 6 3 9 7 9 5 6 1 8 1 2 3 4 0 4 8 0 2 3 1 5 4 2 9 8 4 6 8 0 3 3 3 5 2 8 2 5 2 7 1 5 2 0 7 8 0 1 2 4 5 3 3 4 0 6 9 2 3 3 1 3

Table 4: VAE samples on MNIST.

### A.1.2 CelebA



Table 5: VAE samples on CelebA.

## A.2 WAE

### A.2.1 MNIST

	Latent Dim 8	Latent Dim 32
SSM	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center; width: 48%;"> <p>0 9 9 8 8 6 9 6 0</p> <p>0 8 7 3 7 3 8 0 9 8</p> <p>5 1 8 6 9 9 1 1 0 2</p> <p>8 7 2 6 2 7 2 7 7 1</p> <p>3 0 2 5 8 8 0 3 1 1</p> <p>9 5 9 2 8 6 9 7 3 7</p> <p>4 1 9 1 6 3 9 1 8 6</p> <p>6 4 7 0 1 6 4 3 4 2</p> <p>9 8 0 4 4 7 3 7 3 1</p> <p>1 0 8 9 2 8 6 9 8 6 4</p> </div> <div style="text-align: center; width: 48%;"> <p>6 8 1 1 4 2 5 2 9 6</p> <p>9 1 9 8 9 1 2 7 6 8</p> <p>7 9 2 8 0 4 8 1 2 9</p> <p>2 3 9 4 1 8 7 3 7 0</p> <p>5 1 0 9 6 6 4 2 7 9</p> <p>0 1 7 4 9 3 8 8 6 3</p> <p>1 8 9 2 0 8 3 0 0 9</p> <p>9 8 9 6 9 3 5 0 5 3</p> <p>1 6 3 8 3 6 9 2 1 6</p> <p>8 8 3 8 5 5 0 9 3 3</p> </div> </div>	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center; width: 48%;"> <p>1 8 4 8 5 8 8 3 7 7</p> <p>1 9 6 0 8 8 8 5 5 0</p> <p>7 8 9 7 4 4 8 9 7 4</p> <p>8 3 3 7 1 2 1 1 6 8</p> <p>9 6 2 7 0 0 9 9 5 8</p> <p>5 9 9 6 8 0 6 8 1 8</p> <p>3 9 9 8 2 8 3 3 0 3</p> <p>5 8 6 9 8 0 0 1 8 8</p> <p>9 0 6 8 5 6 9 7 8 9</p> <p>4 1 4 6 6 3 9 9 1 0</p> </div> <div style="text-align: center; width: 48%;"> <p>4 2 9 3 6 9 0 2 1 9</p> <p>6 7 4 8 3 4 9 3 4 9</p> <p>8 3 2 8 9 6 6 7 6 5</p> <p>3 3 5 8 7 2 3 9 6 9</p> <p>8 5 9 9 8 8 0 9 7 3</p> <p>5 9 8 5 4 1 3 0 5 1</p> <p>4 4 5 3 3 2 2 9 0 9</p> <p>5 6 5 0 9 8 9 8 9 8</p> <p>4 4 7 2 0 0 4 0 2 8</p> <p>3 9 3 9 7 0 0 1 3 5</p> </div> </div>
Stein	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center; width: 48%;"> <p>4 3 9 3 8 1 3 2 7 7</p> <p>2 8 3 4 5 4 8 3 3 6</p> <p>5 2 4 1 4 0 0 6 7 4</p> <p>8 0 2 7 2 0 1 2 1 6</p> <p>3 7 8 3 6 6 9 9 3 6</p> <p>0 5 0 6 2 3 0 8 1 8</p> <p>3 9 9 0 9 0 3 7 9 3</p> <p>3 8 9 8 0 1 3 7 2 3</p> <p>9 6 4 8 3 5 9 7 2 4</p> <p>8 7 9 4 6 3 5 3 7 6</p> </div> <div style="text-align: center; width: 48%;"> <p>7 0 3 2 8 9 0 2 1 3</p> <p>6 5 0 8 3 8 9 3 4 9</p> <p>3 7 2 8 9 2 6 7 6 3</p> <p>3 1 4 4 1 6 1 9 4 9</p> <p>8 5 0 8 3 0 0 9 3 7</p> <p>0 9 2 5 4 6 3 0 0 1</p> <p>5 8 3 8 3 2 0 1 0 6</p> <p>4 5 3 8 3 8 9 8 9 1</p> <p>4 2 7 8 5 0 3 0 2 8</p> <p>2 4 3 8 3 9 1 0 3 5</p> </div> </div>	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center; width: 48%;"> <p>4 3 9 3 8 1 3 2 7 7</p> <p>2 8 3 4 5 4 8 3 3 6</p> <p>5 2 4 1 4 0 0 6 7 4</p> <p>8 0 2 7 2 0 1 2 1 6</p> <p>3 7 8 3 6 6 9 9 3 6</p> <p>0 5 0 6 2 3 0 8 1 8</p> <p>3 9 9 0 9 0 3 7 9 3</p> <p>3 8 9 8 0 1 3 7 2 3</p> <p>9 6 4 8 3 5 9 7 2 4</p> <p>8 7 9 4 6 3 5 3 7 6</p> </div> <div style="text-align: center; width: 48%;"> <p>7 0 3 2 8 9 0 2 1 3</p> <p>6 5 0 8 3 8 9 3 4 9</p> <p>3 7 2 8 9 2 6 7 6 3</p> <p>3 1 4 4 1 6 1 9 4 9</p> <p>8 5 0 8 3 0 0 9 3 7</p> <p>0 9 2 5 4 6 3 0 0 1</p> <p>5 8 3 8 3 2 0 1 0 6</p> <p>4 5 3 8 3 8 9 8 9 1</p> <p>4 2 7 8 5 0 3 0 2 8</p> <p>2 4 3 8 3 9 1 0 3 5</p> </div> </div>
Spectral	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center; width: 48%;"> <p>4 3 9 3 8 1 3 2 7 7</p> <p>2 8 3 4 5 4 8 3 3 6</p> <p>5 2 4 1 4 0 0 6 7 4</p> <p>8 0 2 7 2 0 1 2 1 6</p> <p>3 7 8 3 6 6 9 9 3 6</p> <p>0 5 0 6 2 3 0 8 1 8</p> <p>3 9 9 0 9 0 3 7 9 3</p> <p>3 8 9 8 0 1 3 7 2 3</p> <p>9 6 4 8 3 5 9 7 2 4</p> <p>8 7 9 4 6 3 5 3 7 6</p> </div> <div style="text-align: center; width: 48%;"> <p>7 0 3 2 8 9 0 2 1 3</p> <p>6 5 0 8 3 8 9 3 4 9</p> <p>3 7 2 8 9 2 6 7 6 3</p> <p>3 1 4 4 1 6 1 9 4 9</p> <p>8 5 0 8 3 0 0 9 3 7</p> <p>0 9 2 5 4 6 3 0 0 1</p> <p>5 8 3 8 3 2 0 1 0 6</p> <p>4 5 3 8 3 8 9 8 9 1</p> <p>4 2 7 8 5 0 3 0 2 8</p> <p>2 4 3 8 3 9 1 0 3 5</p> </div> </div>	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center; width: 48%;"> <p>4 3 9 3 8 1 3 2 7 7</p> <p>2 8 3 4 5 4 8 3 3 6</p> <p>5 2 4 1 4 0 0 6 7 4</p> <p>8 0 2 7 2 0 1 2 1 6</p> <p>3 7 8 3 6 6 9 9 3 6</p> <p>0 5 0 6 2 3 0 8 1 8</p> <p>3 9 9 0 9 0 3 7 9 3</p> <p>3 8 9 8 0 1 3 7 2 3</p> <p>9 6 4 8 3 5 9 7 2 4</p> <p>8 7 9 4 6 3 5 3 7 6</p> </div> <div style="text-align: center; width: 48%;"> <p>7 0 3 2 8 9 0 2 1 3</p> <p>6 5 0 8 3 8 9 3 4 9</p> <p>3 7 2 8 9 2 6 7 6 3</p> <p>3 1 4 4 1 6 1 9 4 9</p> <p>8 5 0 8 3 0 0 9 3 7</p> <p>0 9 2 5 4 6 3 0 0 1</p> <p>5 8 3 8 3 2 0 1 0 6</p> <p>4 5 3 8 3 8 9 8 9 1</p> <p>4 2 7 8 5 0 3 0 2 8</p> <p>2 4 3 8 3 9 1 0 3 5</p> </div> </div>

Table 6: WAE samples on MNIST.

### A.2.2 CelebA

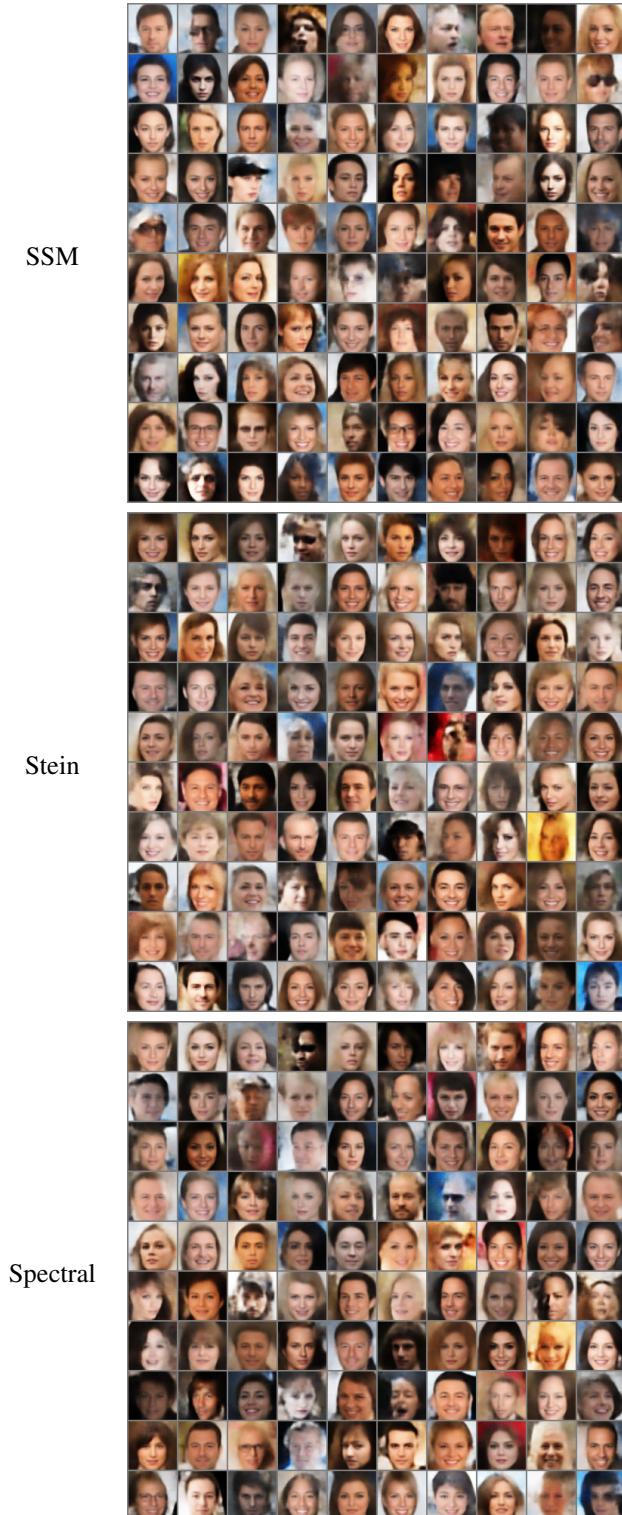


Table 7: WAE samples on CelebA.

## B PROOFS

### B.1 NOTATIONS

Below we provide a summary of the most commonly used notations used in the proofs. First, we denote the data distribution as  $p_d(\mathbf{x})$  and assume that the training/test data  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$  are i.i.d. samples of  $p_d(\mathbf{x})$ . The model is denoted as  $p_m(\mathbf{x}; \boldsymbol{\theta})$ , where  $\boldsymbol{\theta}$  is restricted to a parameter space  $\Theta$ . Note that  $p_m(\mathbf{x}; \mathbf{v})$  can be an unnormalized energy-based model. We use  $\mathbf{v}$  to represent a random vector with the same dimension of input  $\mathbf{x}$ . This vector  $\mathbf{v}$  is often called the projection vector, and we use  $p_{\mathbf{v}}$  to denote its distribution.

Next, we introduce several shorthand notations for quantities related to  $p_m(\mathbf{x}; \boldsymbol{\theta})$  and  $p_d(\mathbf{x})$ . The log-likelihood  $\log p_m(\mathbf{x}; \boldsymbol{\theta})$  and  $\log p_d(\mathbf{x})$  are respectively denoted as  $l_m(\mathbf{x}; \boldsymbol{\theta})$  and  $l_d(\mathbf{x})$ . The (Stein) score function  $\nabla_{\mathbf{x}} \log p_m(\mathbf{x}; \boldsymbol{\theta})$  and  $\nabla_{\mathbf{x}} \log p_d(\mathbf{x})$  are written as  $\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})$  and  $\mathbf{s}_d(\mathbf{x})$ , and finally the Hessian of  $\log p_m(\mathbf{x}; \boldsymbol{\theta})$  w.r.t.  $\mathbf{x}$  is denoted as  $\nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})$ .

We also adopt some convenient notations for collections. In particular, we use  $\mathbf{x}_1^N$  to denote a collection of  $N$  vectors  $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$  and use  $\mathbf{v}_{11}^{NM}$  to denote  $N \times M$  vectors  $\{\mathbf{v}_{11}, \mathbf{v}_{12}, \dots, \mathbf{v}_{1M}, \mathbf{v}_{21}, \mathbf{v}_{22}, \dots, \mathbf{v}_{2M}, \dots, \mathbf{v}_{N1}, \mathbf{v}_{N2}, \dots, \mathbf{v}_{NM}\}$ .

### B.2 BASIC PROPERTIES

The following regularity conditions are needed for integration by parts and identifiability.

**Assumption 1** (Regularity of score functions). *The model score function  $\mathbf{s}_m(\mathbf{x})$  and data score function  $\mathbf{s}_d(\mathbf{x})$  are both differentiable. They additionally satisfy  $\mathbb{E}_{p_d}[\|\mathbf{s}_m(\mathbf{x})\|_2^2] < \infty$  and  $\mathbb{E}_{p_d}[\|\mathbf{s}_d(\mathbf{x})\|_2^2] < \infty$ .*

**Assumption 2** (Regularity of projection vectors). *The projection vectors satisfy  $\mathbb{E}_{p_{\mathbf{v}}}[\|\mathbf{v}\|_2^2] < \infty$ , and  $\mathbb{E}_{p_{\mathbf{v}}}[\mathbf{v}\mathbf{v}^T] \succ 0$ .*

**Assumption 3** (Boundary conditions).  $\forall \boldsymbol{\theta} \in \Theta, \lim_{\|\mathbf{x}\| \rightarrow \infty} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) p_d(\mathbf{x}) = 0$ .

**Assumption 4** (Identifiability). *The model family  $\{p_m(\mathbf{x}; \boldsymbol{\theta}) \mid \boldsymbol{\theta} \in \Theta\}$  is well-specified, i.e.,  $p_d(\mathbf{x}) = p_m(\mathbf{x}; \boldsymbol{\theta}^*)$ . Furthermore,  $p_m(\mathbf{x}; \boldsymbol{\theta}) \neq p_m(\mathbf{x}; \boldsymbol{\theta}^*)$  whenever  $\boldsymbol{\theta} \neq \boldsymbol{\theta}^*$ .*

**Assumption 5** (Positiveness).  $p_m(\mathbf{x}; \boldsymbol{\theta}) > 0, \forall \boldsymbol{\theta} \in \Theta$  and  $\forall \mathbf{x}$ .

**Theorem 1.** *Assume  $\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})$ ,  $\mathbf{s}_d(\mathbf{x})$  and  $p_{\mathbf{v}}$  satisfy some regularity conditions (Assumption 1, Assumption 2). Under proper boundary conditions (Assumption 3), we have*

$$L(\boldsymbol{\theta}; p_{\mathbf{v}}) \triangleq \frac{1}{2} \mathbb{E}_{p_{\mathbf{v}}} \mathbb{E}_{p_d} [(\mathbf{v}^T \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) - \mathbf{v}^T \mathbf{s}_d(\mathbf{x}))^2] = \mathbb{E}_{p_{\mathbf{v}}} \mathbb{E}_{p_d} \left[ \mathbf{v}^T \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v} + \frac{1}{2} (\mathbf{v}^T \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))^2 \right] + C, \quad (13)$$

where  $C$  is a constant w.r.t.  $\boldsymbol{\theta}$ .

*Proof.* The basic idea of this proof is similar to that of Theorem 1 in Hyvärinen (2005). First, note that  $L(\boldsymbol{\theta}, p_{\mathbf{v}})$  can be expanded to

$$\begin{aligned} L(\boldsymbol{\theta}, p_{\mathbf{v}}) &= \frac{1}{2} \mathbb{E}_{p_{\mathbf{v}}} \mathbb{E}_{p_d} [(\mathbf{v}^T \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) - \mathbf{v}^T \mathbf{s}_d(\mathbf{x}))^2] \\ &\stackrel{(i)}{=} \frac{1}{2} \mathbb{E}_{p_{\mathbf{v}}} \mathbb{E}_{p_d} [(\mathbf{v}^T \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))^2 + (\mathbf{v}^T \mathbf{s}_d(\mathbf{x}))^2 - 2(\mathbf{v}^T \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))(\mathbf{v}^T \mathbf{s}_d(\mathbf{x}; \boldsymbol{\theta}))] \end{aligned} \quad (14)$$

$$= \mathbb{E}_{p_{\mathbf{v}}} \mathbb{E}_{p_d} \left[ -(\mathbf{v}^T \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))(\mathbf{v}^T \mathbf{s}_d(\mathbf{x}; \boldsymbol{\theta})) + \frac{1}{2} (\mathbf{v}^T \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))^2 \right] + C, \quad (15)$$

where (i) is due to the assumptions of bounded expectations. We have absorbed the second term in the bracket of (14) into  $C$  since it does not depend on  $\boldsymbol{\theta}$ . Now what we need to prove is

$$-\mathbb{E}_{p_{\mathbf{v}}} \mathbb{E}_{p_d} [(\mathbf{v}^T \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))(\mathbf{v}^T \mathbf{s}_d(\mathbf{x}; \boldsymbol{\theta}))] = \mathbb{E}_{p_{\mathbf{v}}} \mathbb{E}_{p_d} [\mathbf{v}^T \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v}]. \quad (16)$$

This can be shown by first observing that

$$-\mathbb{E}_{p_{\mathbf{v}}} \mathbb{E}_{p_d} [(\mathbf{v}^T \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))(\mathbf{v}^T \mathbf{s}_d(\mathbf{x}; \boldsymbol{\theta}))]$$

$$\begin{aligned}
&= -\mathbb{E}_{p_{\mathbf{v}}} \int p_d(\mathbf{x})(\mathbf{v}^\top \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))(\mathbf{v}^\top \mathbf{s}_d(\mathbf{x}; \boldsymbol{\theta})) d\mathbf{x} \\
&= -\mathbb{E}_{p_{\mathbf{v}}} \int p_d(\mathbf{x})(\mathbf{v}^\top \nabla_{\mathbf{x}} \log p_m(\mathbf{x}; \boldsymbol{\theta}))(\mathbf{v}^\top \nabla_{\mathbf{x}} \log p_d(\mathbf{x})) d\mathbf{x} \\
&= -\mathbb{E}_{p_{\mathbf{v}}} \int (\mathbf{v}^\top \nabla_{\mathbf{x}} \log p_m(\mathbf{x}; \boldsymbol{\theta}))(\mathbf{v}^\top \nabla_{\mathbf{x}} p_d(\mathbf{x})) d\mathbf{x} \\
&= -\mathbb{E}_{p_{\mathbf{v}}} \sum_{i=1}^D \int (\mathbf{v}^\top \nabla_{\mathbf{x}} \log p_m(\mathbf{x}; \boldsymbol{\theta})) v_i \frac{\partial p_d(\mathbf{x})}{\partial x_i} d\mathbf{x},
\end{aligned} \tag{17}$$

where we assume  $\mathbf{x} \in \mathbb{R}^D$ . Then, applying multivariate integration by parts (cf., Lemma 4 in Hyvärinen (2005)), we obtain

$$\begin{aligned}
&\left| \mathbb{E}_{p_{\mathbf{v}}} \sum_{i=1}^D \int (\mathbf{v}^\top \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})) v_i \frac{\partial p_d(\mathbf{x})}{\partial x_i} d\mathbf{x} + \mathbb{E}_{p_{\mathbf{v}}} \sum_{i=1}^D \int v_i p_d(\mathbf{x}) \mathbf{v}^\top \frac{\partial \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})}{\partial x_i} d\mathbf{x} \right| \\
&= \left| \mathbb{E}_{p_{\mathbf{v}}} \left[ \sum_{i=1}^D \lim_{x_i \rightarrow \infty} (\mathbf{v}^\top \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})) v_i p_d(\mathbf{x}) - \sum_{i=1}^D \lim_{x_i \rightarrow -\infty} (\mathbf{v}^\top \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})) v_i p_d(\mathbf{x}) \right] \right| \\
&\leq \sum_{i=1}^D \lim_{x_i \rightarrow \infty} \sum_{j=1}^D \mathbb{E}_{p_{\mathbf{v}}} |v_i v_j| |s_{m,j}(\mathbf{x}; \boldsymbol{\theta}) p_d(\mathbf{x})| + \sum_{i=1}^D \lim_{x_i \rightarrow -\infty} \sum_{j=1}^D \mathbb{E}_{p_{\mathbf{v}}} |v_i v_j| |s_{m,j}(\mathbf{x}; \boldsymbol{\theta}) p_d(\mathbf{x})| \\
&\stackrel{(i)}{\leq} \sum_{i=1}^D \lim_{x_i \rightarrow \infty} \sum_{j=1}^D \sqrt{\mathbb{E}_{p_{\mathbf{v}}} v_i^2 \mathbb{E}_{p_{\mathbf{v}}} v_j^2} |s_{m,j}(\mathbf{x}; \boldsymbol{\theta}) p_d(\mathbf{x})| + \sum_{i=1}^D \lim_{x_i \rightarrow -\infty} \sum_{j=1}^D \sqrt{\mathbb{E}_{p_{\mathbf{v}}} v_i^2 \mathbb{E}_{p_{\mathbf{v}}} v_j^2} |s_{m,j}(\mathbf{x}; \boldsymbol{\theta}) p_d(\mathbf{x})| \\
&\stackrel{(ii)}{=} 0,
\end{aligned}$$

where  $s_{m,j}(\mathbf{x}; \boldsymbol{\theta})$  denotes the  $j$ -th component of  $\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})$ . In the above derivation, (i) is due to Cauchy-Schwarz inequality and (ii) is from the assumption that  $\mathbb{E}_{p_{\mathbf{v}}} [\|\mathbf{v}\|^2] < \infty$  and  $s(\mathbf{x}; \boldsymbol{\theta}) p_d(\mathbf{x})$  vanishes at infinity.

Now returning to (17), we have

$$\begin{aligned}
-\mathbb{E}_{p_{\mathbf{v}}} \sum_{i=1}^D \int (\mathbf{v}^\top \nabla_{\mathbf{x}} \log p_m(\mathbf{x}; \boldsymbol{\theta})) v_i \frac{\partial p_d(\mathbf{x})}{\partial x_i} d\mathbf{x} &= \mathbb{E}_{p_{\mathbf{v}}} \sum_{i=1}^D \int v_i p_d(\mathbf{x}) \mathbf{v}^\top \frac{\partial \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})}{\partial x_i} d\mathbf{x} \\
&= \mathbb{E}_{p_{\mathbf{v}}} \int p_d(\mathbf{x}) \mathbf{v}^\top \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v} d\mathbf{x},
\end{aligned}$$

which proves (16) and the proof is completed.  $\square$

**Lemma 1.** Assume our model family is well-specified and identifiable (Assumption 4). Assume further that the densities are all positive (Assumption 5). When  $p_{\mathbf{v}}$  satisfies some regularity conditions (Assumption 2), we have

$$L(\boldsymbol{\theta}; p_{\mathbf{v}}) = 0 \Leftrightarrow \boldsymbol{\theta} = \boldsymbol{\theta}^*.$$

*Proof.* First, since  $p_d(\mathbf{x}) = p_m(\mathbf{x}; \boldsymbol{\theta}^*) > 0$ ,  $L(\boldsymbol{\theta}; p_{\mathbf{v}}) = 0$  implies

$$\begin{aligned}
&\frac{1}{2} \mathbb{E}_{p_{\mathbf{v}}} (\mathbf{v}^\top (\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) - \mathbf{s}_d(\mathbf{x})))^2 = 0 \\
&\Leftrightarrow \mathbb{E}_{p_{\mathbf{v}}} \mathbf{v}^\top (\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) - \mathbf{s}_d(\mathbf{x})) (\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) - \mathbf{s}_d(\mathbf{x}))^\top \mathbf{v} = 0 \\
&\Leftrightarrow (\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) - \mathbf{s}_d(\mathbf{x}))^\top \mathbb{E}_{p_{\mathbf{v}}} [\mathbf{v} \mathbf{v}^\top] (\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) - \mathbf{s}_d(\mathbf{x})) = 0 \\
&\stackrel{(i)}{\Leftrightarrow} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) - \mathbf{s}_d(\mathbf{x}) = 0 \\
&\Leftrightarrow \log p_m(\mathbf{x}; \boldsymbol{\theta}) = \log p_d(\mathbf{x}) + C
\end{aligned}$$

where (i) holds because  $\mathbb{E}_{p_{\mathbf{v}}} [\mathbf{v} \mathbf{v}^\top]$  is positive definite. Because  $p_m(\mathbf{x}; \boldsymbol{\theta})$  and  $p_d(\mathbf{x})$  are normalized probability density functions, we have  $p_m(\mathbf{x}; \boldsymbol{\theta}) = p_d(\mathbf{x})$ . The identifiability assumption gives  $\boldsymbol{\theta} = \boldsymbol{\theta}^*$ . This concludes the left to right direction of the implication and the converse direction is trivial.  $\square$

### B.3 CONSISTENCY

In addition to the assumptions in Theorem 1 and Lemma 1, we need the following regularity conditions to prove the consistency of  $\hat{\theta}_{N,M} \triangleq \arg \min_{\theta \in \Theta} \hat{J}(\theta; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM})$ .

**Assumption 6** (Compactness). *The parameter space  $\Theta$  is compact.*

**Assumption 7** (Lipschitz continuity). *Both  $\nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \theta)$  and  $\mathbf{s}_m(\mathbf{x}; \theta) \mathbf{s}_m(\mathbf{x}; \theta)^T$  are Lipschitz continuous in terms of Frobenius norm, i.e.,  $\forall \theta_1 \in \Theta, \theta_2 \in \Theta$ ,  $\|\nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \theta_1) - \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \theta_2)\|_F \leq L_1(\mathbf{x}) \|\theta_1 - \theta_2\|_2$  and  $\|\mathbf{s}_m(\mathbf{x}; \theta_1) \mathbf{s}_m(\mathbf{x}; \theta_1)^T - \mathbf{s}_m(\mathbf{x}; \theta_2) \mathbf{s}_m(\mathbf{x}; \theta_2)^T\|_F \leq L_2(\mathbf{x}) \|\theta_1 - \theta_2\|_2$ . In addition, we require  $\mathbb{E}_{p_d}[L_1^2(\mathbf{x})] < \infty$  and  $\mathbb{E}_{p_d}[L_2^2(\mathbf{x})] < \infty$ .*

**Assumption 8** (Bounded moments of projection vectors).  $\mathbb{E}_{p_{\mathbf{v}}}[\|\mathbf{v}\mathbf{v}^T\|_F^2] < \infty$ .

**Lemma 2.** *Suppose  $\mathbf{s}_m(\mathbf{x}; \theta)$  is sufficiently smooth (Assumption 7) and  $p_{\mathbf{v}}$  has bounded higher-order moments (Assumption 8). Let  $f(\theta; \mathbf{x}, \mathbf{v}) \triangleq \mathbf{v}^T \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \theta) \mathbf{v} + \frac{1}{2}(\mathbf{v}^T \mathbf{s}_m(\mathbf{x}; \theta))^2$ . Then  $f(\theta; \mathbf{x}, \mathbf{v})$  is Lipschitz continuous with constant  $L(\mathbf{x}, \mathbf{v})$  and  $\mathbb{E}_{p_d, p_{\mathbf{v}}}[L^2(\mathbf{x}, \mathbf{v})] < \infty$ .*

*Proof.* Let  $A(\theta) \triangleq \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \theta)$  and  $B(\theta) \triangleq \mathbf{s}_m(\mathbf{x}; \theta) \mathbf{s}_m(\mathbf{x}; \theta)^T$ . Consider  $\theta_1 \in \Theta, \theta_2 \in \Theta$  and let  $D$  be the dimension of  $\mathbf{v}$ , we have

$$\begin{aligned} & |f(\theta_1; \mathbf{x}, \mathbf{v}) - f(\theta_2; \mathbf{x}, \mathbf{v})| \\ &= \sum_{i=1}^D \sum_{j=1}^D \left[ v_i v_j (A(\theta_1)_{ij} - A(\theta_2)_{ij}) + \frac{1}{2} v_i v_j (B(\theta_1)_{ij} - B(\theta_2)_{ij}) \right] \\ &\stackrel{(i)}{\leq} \sqrt{\sum_{i=1}^D \sum_{j=1}^D v_i^2 v_j^2} \sqrt{\sum_{i=1}^D \sum_{j=1}^D \left[ (A(\theta_1)_{ij} - A(\theta_2)_{ij}) + \frac{1}{2} (B(\theta_1)_{ij} - B(\theta_2)_{ij}) \right]^2} \\ &\stackrel{(ii)}{\leq} \sqrt{\sum_{i=1}^D \sum_{j=1}^D v_i^2 v_j^2} \sqrt{\sum_{i=1}^D \sum_{j=1}^D 2(A(\theta_1)_{ij} - A(\theta_2)_{ij})^2 + \frac{1}{2} (B(\theta_1)_{ij} - B(\theta_2)_{ij})^2} \\ &\stackrel{(iii)}{\leq} \sqrt{\sum_{i=1}^D \sum_{j=1}^D v_i^2 v_j^2} \sqrt{2L_1^2(\mathbf{x}) \|\theta_1 - \theta_2\|_2^2 + \frac{1}{2} L_2^2(\mathbf{x}) \|\theta_1 - \theta_2\|_2^2} \\ &= \sqrt{\sum_{i=1}^D \sum_{j=1}^D v_i^2 v_j^2} \sqrt{2L_1^2(\mathbf{x}) + \frac{1}{2} L_2^2(\mathbf{x})} \|\theta_1 - \theta_2\|_2, \end{aligned}$$

where (i) is Cauchy-Schwarz inequality, (ii) is Jensen's inequality, and (iii) is due to Assumption 7. Now let

$$L(\mathbf{x}, \mathbf{v}) \triangleq \sqrt{\sum_{i=1}^D \sum_{j=1}^D v_i^2 v_j^2} \sqrt{2L_1^2(\mathbf{x}) + \frac{1}{2} L_2^2(\mathbf{x})}.$$

Then

$$\begin{aligned} \mathbb{E}_{p_d, p_{\mathbf{v}}}[L^2(\mathbf{x}, \mathbf{v})] &\stackrel{(i)}{=} \mathbb{E}_{p_{\mathbf{v}}} \left[ \sum_{i=1}^D \sum_{j=1}^D v_i^2 v_j^2 \right] \mathbb{E}_{p_d} \left[ 2L_1^2(\mathbf{x}) + \frac{1}{2} L_2^2(\mathbf{x}) \right] \\ &\stackrel{(ii)}{<} \infty, \end{aligned}$$

where (i) results from the independence of  $\mathbf{v}, \mathbf{x}$ , and (ii) is due to Assumption 7 and Assumption 8.  $\square$

**Lemma 3** (Uniform convergence of the expected error). *Under Assumption 6-8, we have*

$$\mathbb{E}_{p_{\mathbf{v}}, p_d} \left[ \sup_{\theta \in \Theta} |\hat{J}(\theta; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) - J(\theta; p_{\mathbf{v}})| \right] \leq O \left( \text{diam}(\Theta) \sqrt{\frac{D}{N}} \right) \quad (18)$$

where  $\text{diam}(\cdot)$  denotes the diameter and  $D$  is the dimension of  $\Theta$ .

*Proof.* The proof consists of 3 steps. First, we use the symmetrization trick to get rid of the inner expectation term  $J(\boldsymbol{\theta}; p_{\mathbf{v}}) = \mathbb{E}_{p_{\mathbf{v}}, p_d}[\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM})]$ . Second, we use chaining to get an upper bound that involves integration of the metric entropy. Finally, we upper bound the metric entropy to obtain the uniform convergence bound.

**Step 1:** From Jensen's inequality, we obtain

$$\begin{aligned} & \mathbb{E}_{p_{\mathbf{v}}, p_d} \left[ \sup_{\boldsymbol{\theta} \in \Theta} |\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) - J(\boldsymbol{\theta}; p_{\mathbf{v}})| \right] \\ &= \mathbb{E}_{p_{\mathbf{v}}, p_d} \left[ \sup_{\boldsymbol{\theta} \in \Theta} |\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) - \mathbb{E}_{p_{\mathbf{v}}, p_d}[\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM})]| \right] \\ &= \mathbb{E}_{p_{\mathbf{v}}, p_d} \left[ \sup_{\boldsymbol{\theta} \in \Theta} |\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) - \mathbb{E}_{p_{\mathbf{v}}, p_d}[\hat{J}(\boldsymbol{\theta}; \mathbf{x}'_1^N, \mathbf{v}'_{11}^{NM})]| \right] \\ &\leq \mathbb{E}_{p_{\mathbf{v}}, p_d} \left[ \sup_{\boldsymbol{\theta} \in \Theta} |\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) - \hat{J}(\boldsymbol{\theta}; \mathbf{x}'_1^N, \mathbf{v}'_{11}^{NM})| \right], \end{aligned}$$

where  $\mathbf{x}'_1^N, \mathbf{v}'_{11}^{NM}$  are independent copies of  $\mathbf{x}_1^N$  and  $\mathbf{v}_{11}^{NM}$ . Let  $\{\epsilon_i\}_{i=1}^N$  be a set of independent Rademacher random variables, we have

$$\begin{aligned} & \mathbb{E}_{p_{\mathbf{v}}, p_d} \left[ \sup_{\boldsymbol{\theta} \in \Theta} |\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) - \hat{J}(\boldsymbol{\theta}; \mathbf{x}'_1^N, \mathbf{v}'_{11}^{NM})| \right] \\ &= \mathbb{E}_{p_{\mathbf{v}}, p_d} \left[ \sup_{\boldsymbol{\theta} \in \Theta} \frac{1}{N} \frac{1}{M} \left| \sum_{i=1}^N \sum_{j=1}^M \underbrace{\mathbf{v}_{ij}^\top \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}_i; \boldsymbol{\theta}) \mathbf{v}_{ij} + \frac{1}{2} (\mathbf{v}_{ij}^\top \mathbf{s}_m(\mathbf{x}_i; \boldsymbol{\theta}))^2}_{\triangleq f(\boldsymbol{\theta}; \mathbf{x}_i, \mathbf{v}_{ij})} - \underbrace{\mathbf{v}'_{ij}^\top \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}'_i; \boldsymbol{\theta}) \mathbf{v}'_{ij} + \frac{1}{2} (\mathbf{v}'_{ij}^\top \mathbf{s}_m(\mathbf{x}'_i; \boldsymbol{\theta}))^2}_{\triangleq f(\boldsymbol{\theta}; \mathbf{x}'_i, \mathbf{v}'_{ij})} \right| \right] \\ &= \mathbb{E}_{p_{\mathbf{v}}, p_d} \left[ \sup_{\boldsymbol{\theta} \in \Theta} \frac{1}{N} \frac{1}{M} \left| \sum_{i=1}^N \sum_{j=1}^M f(\boldsymbol{\theta}; \mathbf{x}_i, \mathbf{v}_{ij}) - f(\boldsymbol{\theta}; \mathbf{x}'_i, \mathbf{v}'_{ij}) \right| \right] \\ &\stackrel{(i)}{=} \mathbb{E} \left[ \sup_{\boldsymbol{\theta} \in \Theta} \left| \frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M \epsilon_i (f(\boldsymbol{\theta}; \mathbf{x}_i, \mathbf{v}_{ij}) - f(\boldsymbol{\theta}; \mathbf{x}'_i, \mathbf{v}'_{ij})) \right| \right] \\ &\stackrel{(ii)}{\leq} \mathbb{E} \left[ \sup_{\boldsymbol{\theta} \in \Theta} \left| \frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M \epsilon_i f(\boldsymbol{\theta}; \mathbf{x}_i, \mathbf{v}_{ij}) \right| \right] + \mathbb{E} \left[ \sup_{\boldsymbol{\theta} \in \Theta} \left| \frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M \epsilon_i f(\boldsymbol{\theta}; \mathbf{x}'_i, \mathbf{v}'_{ij}) \right| \right] \\ &= 2 \mathbb{E} \left[ \sup_{\boldsymbol{\theta} \in \Theta} \left| \frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M \epsilon_i f(\boldsymbol{\theta}; \mathbf{x}_i, \mathbf{v}_{ij}) \right| \right], \tag{19} \end{aligned}$$

where (i) is because the quantity is symmetric about 0 in distribution, and (ii) is due to Jensen's inequality.

**Step 2:** First note that given  $\mathbf{x}_i, \mathbf{v}_{ij}$ ,  $\frac{\epsilon_i}{M} \sum_{j=1}^M f(\boldsymbol{\theta}; \mathbf{x}_i, \mathbf{v}_{ij})$  is a zero-mean sub-Gaussian process w.r.t.  $\boldsymbol{\theta}$ . This can be observed from

$$\begin{aligned} & \mathbb{E}_{\epsilon_i} \left[ e^{\lambda \frac{\epsilon_i}{M} \sum_{j=1}^M [f(\boldsymbol{\theta}_1; \mathbf{x}_i, \mathbf{v}_{ij}) - f(\boldsymbol{\theta}_2; \mathbf{x}_i, \mathbf{v}_{ij})]} \right] \\ &\stackrel{(i)}{\leq} \exp \left\{ \frac{\lambda^2}{2M^2} \left( \sum_{j=1}^M [f(\boldsymbol{\theta}_1; \mathbf{x}_i, \mathbf{v}_{ij}) - f(\boldsymbol{\theta}_2; \mathbf{x}_i, \mathbf{v}_{ij})] \right)^2 \right\} \\ &\stackrel{(ii)}{\leq} \exp \left\{ \frac{\lambda^2}{2M^2} M \sum_{j=1}^M [f(\boldsymbol{\theta}_1; \mathbf{x}_i, \mathbf{v}_{ij}) - f(\boldsymbol{\theta}_2; \mathbf{x}_i, \mathbf{v}_{ij})]^2 \right\} \\ &\stackrel{(iii)}{\leq} \exp \left\{ \frac{\lambda^2}{2M} \sum_{j=1}^M L^2(\mathbf{x}_i, \mathbf{v}_{ij}) \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|^2 \right\}, \end{aligned}$$

where (i) holds because  $\epsilon_i$  is a 1-sub-Gaussian random variable, (ii) is from Cauchy-Schwarz inequality and (iii) is due to Lemma 2. As a result,  $\frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M \epsilon_i f(\boldsymbol{\theta}; \mathbf{x}_i, \mathbf{v}_{ij})$  is a zero-mean sub-Gaussian random process with

metric

$$d(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2) = \frac{1}{\sqrt{N}} \sqrt{\frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M L^2(\mathbf{x}_i, \mathbf{v}_{ij})} \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|.$$

Since  $\Theta$  is compact, the diameter of  $\Theta$  with respect to the Euclidean norm  $\|\cdot\|_2$  is finite and we denote it as  $\text{diam}(\Theta) < \infty$ . Then, Dudley's entropy integral (Dudley, 1967) gives

$$\mathbb{E} \left[ \sup_{\boldsymbol{\theta} \in \Theta} \left| \frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M \epsilon_i f(\boldsymbol{\theta}; \mathbf{x}_i, \mathbf{v}_{ij}) \right| \right] \leq O(1) \mathbb{E} \left[ \int_0^{\frac{1}{\sqrt{N}} \sqrt{\frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M L^2(\mathbf{x}_i, \mathbf{v}_{ij})} \text{diam}(\Theta)} \sqrt{\log N(\Theta, d, \epsilon)} d\epsilon \right]. \quad (20)$$

Here  $\frac{\log N(\Theta, d, \epsilon)}{\frac{1}{\sqrt{N}} \sqrt{\frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M L^2(\mathbf{x}_i, \mathbf{v}_{ij})} \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2}$  is the metric entropy of  $\Theta$  with metric  $d(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2) = \sqrt{\frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M L^2(\mathbf{x}_i, \mathbf{v}_{ij})}$  and size  $\epsilon$ .

**Step 3:** When the dimension of  $\boldsymbol{\theta} \in \Theta$  is  $D$ , it is known that the  $\epsilon$ -covering number of  $\Theta$  with Euclidean distance is

$$N(\Theta, \|\cdot\|, \epsilon) \leq \left( 1 + \frac{\text{diam}(\Theta)}{\epsilon} \right)^D.$$

Therefore,  $N(\Theta, d, \epsilon)$  can be bounded by

$$N(\Theta, d, \epsilon) \leq \left( 1 + \sqrt{\frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M L^2(\mathbf{x}_i, \mathbf{v}_{ij})} \frac{\text{diam}(\Theta)}{\sqrt{N}\epsilon} \right)^D.$$

Hence, the metric integral can be bounded

$$\begin{aligned} & \int_0^{\frac{1}{\sqrt{N}} \sqrt{\frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M L^2(\mathbf{x}_i, \mathbf{v}_{ij})} \text{diam}(\Theta)} \sqrt{\log N(\Theta, d, \epsilon)} d\epsilon \\ & \leq \int_0^{\frac{1}{\sqrt{N}} \sqrt{\frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M L^2(\mathbf{x}_i, \mathbf{v}_{ij})} \text{diam}(\Theta)} \sqrt{D \log \left( 1 + \sqrt{\frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M L^2(\mathbf{x}_i, \mathbf{v}_{ij})} \frac{\text{diam}(\Theta)}{\sqrt{N}\epsilon} \right)} d\epsilon \\ & \leq \int_0^{\frac{1}{\sqrt{N}} \sqrt{\frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M L^2(\mathbf{x}_i, \mathbf{v}_{ij})} \text{diam}(\Theta)} \sqrt{\sqrt{\frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M L^2(\mathbf{x}_i, \mathbf{v}_{ij})} \frac{D \text{diam}(\Theta)}{\sqrt{N}\epsilon}} d\epsilon \\ & = 2 \sqrt{\frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M L^2(\mathbf{x}_i, \mathbf{v}_{ij})} \sqrt{\frac{D}{N}} \text{diam}(\Theta) \end{aligned} \quad (21)$$

Finally, combining (19), (20) and (21) gives us

$$\begin{aligned} & \mathbb{E}_{p_{\mathbf{v}}, p_d} \left[ \sup_{\boldsymbol{\theta} \in \Theta} |\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) - J(\boldsymbol{\theta}; p_{\mathbf{v}})| \right] \\ & \leq 4O(1) \mathbb{E}_{p_{\mathbf{v}}, p_d} \left[ \sqrt{\frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M L^2(\mathbf{x}_i, \mathbf{v}_{ij})} \sqrt{\frac{D}{N}} \text{diam}(\Theta) \right] \\ & \stackrel{(i)}{\leq} O(1) \sqrt{\frac{D}{N}} \text{diam}(\Theta) \sqrt{\mathbb{E}_{p_{\mathbf{v}}, p_d} \left[ \frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M L^2(\mathbf{x}_i, \mathbf{v}_{ij}) \right]} \end{aligned}$$

$$\stackrel{(ii)}{\leq} O(1) \operatorname{diam}(\Theta) \sqrt{\frac{D}{N}},$$

where (i) is due to Jensen's inequality and (ii) results from  $\mathbb{E}[L^2(\mathbf{x}, \mathbf{v})] < \infty$ , and the compactness of  $\Theta$  guarantees that the bound is finite.  $\square$

**Theorem 2.** Suppose all the previous assumptions hold (Assumption 1-Assumption 8). Assume further the conditions of Theorem 1 and Lemma 1 are satisfied. Let  $\boldsymbol{\theta}^*$  be the true parameter of the data distribution, and  $\hat{\boldsymbol{\theta}}_{N,M}$  be the empirical estimator defined by

$$\hat{\boldsymbol{\theta}}_{N,M} \triangleq \arg \min_{\boldsymbol{\theta} \in \Theta} \hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM})$$

Then,  $\hat{\boldsymbol{\theta}}_{N,M}$  is consistent, meaning that

$$\hat{\boldsymbol{\theta}}_{N,M} \xrightarrow{p} \boldsymbol{\theta}^*$$

as  $N \rightarrow \infty$ .

*Proof.* Note that Theorem 1 and Lemma 1 together imply that  $\boldsymbol{\theta}^* = \arg \min_{\boldsymbol{\theta} \in \Theta} J(\boldsymbol{\theta}; p_{\mathbf{v}})$ . Then, we will show  $J(\hat{\boldsymbol{\theta}}_{N,M}; p_{\mathbf{v}}) \xrightarrow{p} J(\boldsymbol{\theta}^*; p_{\mathbf{v}})$  when  $N \rightarrow \infty$ . This can be done by noticing

$$\begin{aligned} J(\hat{\boldsymbol{\theta}}_{N,M}; p_{\mathbf{v}}) - J(\boldsymbol{\theta}^*; p_{\mathbf{v}}) &= J(\hat{\boldsymbol{\theta}}_{N,M}; p_{\mathbf{v}}) - \hat{J}(\hat{\boldsymbol{\theta}}_{N,M}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) + \hat{J}(\hat{\boldsymbol{\theta}}_{N,M}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) - \hat{J}(\boldsymbol{\theta}^*; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) \\ &\quad + \hat{J}(\boldsymbol{\theta}^*; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) - J(\boldsymbol{\theta}^*; p_{\mathbf{v}}) \\ &\leq \sup_{\boldsymbol{\theta} \in \Theta} |\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) - J(\boldsymbol{\theta}; p_{\mathbf{v}})| + |\hat{J}(\boldsymbol{\theta}^*; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) - J(\boldsymbol{\theta}^*; p_{\mathbf{v}})| \\ &\leq 2 \sup_{\boldsymbol{\theta} \in \Theta} |\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) - J(\boldsymbol{\theta}; p_{\mathbf{v}})| \end{aligned} \tag{22}$$

We can easily conclude that (22) is  $o_p(1)$  with the help of Lemma 3, because

$$P\left(\sup_{\boldsymbol{\theta} \in \Theta} |\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) - J(\boldsymbol{\theta}; p_{\mathbf{v}})| > t\right) \leq \mathbb{E}\left[\sup_{\boldsymbol{\theta} \in \Theta} |\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) - J(\boldsymbol{\theta}; p_{\mathbf{v}})|\right]/t \leq O(1)\sqrt{\frac{1}{Nt^2}} \rightarrow 0,$$

as  $N \rightarrow \infty$ . From Lemma 1 we also have  $L(\hat{\boldsymbol{\theta}}_{N,M}; p_{\mathbf{v}}) - L(\boldsymbol{\theta}^*; p_{\mathbf{v}}) > 0$  if  $\hat{\boldsymbol{\theta}}_{N,M} \neq \boldsymbol{\theta}$ . As shown by Theorem 1, this is the same as  $J(\hat{\boldsymbol{\theta}}_{N,M}; p_{\mathbf{v}}) - J(\boldsymbol{\theta}^*; p_{\mathbf{v}}) > 0$  if  $\hat{\boldsymbol{\theta}}_{N,M} \neq \boldsymbol{\theta}$ . Therefore (22) =  $o_p(1)$  gives  $J(\hat{\boldsymbol{\theta}}_{N,M}; p_{\mathbf{v}}) \xrightarrow{p} J(\boldsymbol{\theta}^*; p_{\mathbf{v}})$ .

Next, we show  $\hat{\boldsymbol{\theta}}_{N,M} \xrightarrow{p} \boldsymbol{\theta}^*$ . This can be inferred from  $J(\hat{\boldsymbol{\theta}}_{N,M}; p_{\mathbf{v}}) \xrightarrow{p} J(\boldsymbol{\theta}^*; p_{\mathbf{v}})$  because  $J(\boldsymbol{\theta}; p_{\mathbf{v}})$  is continuous (Assumption 7) and  $\Theta$  is compact (Assumption 6). The proof is reductio ad absurdum. Specifically, assume that  $\hat{\boldsymbol{\theta}}_{N,M} \not\xrightarrow{p} \boldsymbol{\theta}^*$ . We know  $\exists \epsilon > 0, \delta > 0, \forall K > 0, \exists N > K, M > 0$  such that  $P(\|\hat{\boldsymbol{\theta}}_{N,M} - \boldsymbol{\theta}^*\| \geq \epsilon) \geq \delta$ . Note that  $J(\boldsymbol{\theta}; p_{\mathbf{v}}) = \mathbb{E}[f(\boldsymbol{\theta}; \mathbf{x}, \mathbf{v})]$ ,  $f(\boldsymbol{\theta}; \mathbf{x}, \mathbf{v})$  is Lipschitz continuous and  $\mathbb{E}[L(\boldsymbol{\theta}; \mathbf{x}, \mathbf{v})] \leq \sqrt{\mathbb{E}[L^2(\boldsymbol{\theta}; \mathbf{x}, \mathbf{v})]} < \infty$ . This implies that  $J(\boldsymbol{\theta}; p_{\mathbf{v}})$  is continuous w.r.t.  $\boldsymbol{\theta}$ . Since  $\Theta$  is compact and  $J(\boldsymbol{\theta}; p_{\mathbf{v}})$  is continuous, we can define a compact set  $\mathcal{S}_\epsilon \triangleq \{\boldsymbol{\theta} \in \Theta \mid \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\| \geq \epsilon\}$  and let  $\boldsymbol{\theta}_{\mathcal{S}_\epsilon} \triangleq \arg \min_{\boldsymbol{\theta} \in \mathcal{S}_\epsilon} J(\boldsymbol{\theta}; p_{\mathbf{v}})$ . Observe that

$$\begin{aligned} p(J(\hat{\boldsymbol{\theta}}_{N,M}; p_{\mathbf{v}}) \geq J(\boldsymbol{\theta}_{\mathcal{S}_\epsilon}; p_{\mathbf{v}})) &= p(|J(\hat{\boldsymbol{\theta}}_{N,M}; p_{\mathbf{v}}) - J(\boldsymbol{\theta}^*; p_{\mathbf{v}})| \geq J(\boldsymbol{\theta}_{\mathcal{S}_\epsilon}; p_{\mathbf{v}}) - J(\boldsymbol{\theta}^*; p_{\mathbf{v}})) \\ &\geq p(\|\hat{\boldsymbol{\theta}}_{N,M} - \boldsymbol{\theta}^*\| \geq \epsilon) \geq \delta. \end{aligned}$$

However, the fact that  $p(|J(\hat{\boldsymbol{\theta}}_{N,M}; p_{\mathbf{v}}) - J(\boldsymbol{\theta}^*; p_{\mathbf{v}})| \geq J(\boldsymbol{\theta}_{\mathcal{S}_\epsilon}; p_{\mathbf{v}}) - J(\boldsymbol{\theta}^*; p_{\mathbf{v}})) \geq \delta$  holds for arbitrarily large  $N$  contradicts  $J(\hat{\boldsymbol{\theta}}_{N,M}; p_{\mathbf{v}}) \xrightarrow{p} J(\boldsymbol{\theta}^*; p_{\mathbf{v}})$ .  $\square$

## B.4 ASYMPTOTIC NORMALITY

**Notations.** To simplify notations we use  $\partial_i \partial_j h(\cdot) \triangleq (\nabla_{\mathbf{x}}^2 h(\cdot))_{ij}$ ,  $\partial_i h(\cdot) \triangleq (\nabla_{\mathbf{x}} h(\cdot))_i$ , and denote  $\nabla_{\mathbf{x}} h(\mathbf{x})|_{\mathbf{x}=\mathbf{x}'}$  as  $\nabla_{\mathbf{x}} h(\mathbf{x}')$ . Here  $h(\cdot)$  denotes some arbitrary function. Let  $l_m \triangleq \log p_m(\mathbf{x}; \boldsymbol{\theta})$ ,  $l_m(\mathbf{x}; \boldsymbol{\theta}) \triangleq \log p_m(\mathbf{x}; \boldsymbol{\theta})$  and further adopt the following notations

$$J(\boldsymbol{\theta}) \triangleq \mathbb{E}_{p_d} \left[ \operatorname{tr}(\nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})) + \frac{1}{2} \|\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})\|_2^2 \right]$$

$$\begin{aligned}
f(\boldsymbol{\theta}; \mathbf{x}, \mathbf{v}) &\triangleq \mathbf{v}^\top \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v} + \frac{1}{2} (\mathbf{v}^\top \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))^2 \\
f(\boldsymbol{\theta}; \mathbf{x}, \mathbf{v}_1^M) &\triangleq \frac{1}{M} \sum_{j=1}^M f(\boldsymbol{\theta}; \mathbf{x}, \mathbf{v}_j) = \frac{1}{M} \sum_{j=1}^M \mathbf{v}_j^\top \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v}_j + \frac{1}{2} (\mathbf{v}_j^\top \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))^2 \\
f(\boldsymbol{\theta}; \mathbf{x}) &\triangleq \text{tr}(\nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})) + \frac{1}{2} \|\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})\|_2^2 \\
\Sigma_{ij} &\triangleq (\mathbb{E}_{p_{\mathbf{v}}}[\mathbf{v}\mathbf{v}^\top])_{ij} \\
\mathfrak{S}_{ijpq} &\triangleq \mathbb{E}_{p_{\mathbf{v}}} [v_i v_j v_p v_q] \\
\mathfrak{V}_{ijpq} &\triangleq \mathbb{E}_{p_d} \left[ \left( \nabla_{\boldsymbol{\theta}} \partial_i \partial_j l_m + \frac{1}{2} \nabla_{\boldsymbol{\theta}} (\partial_i l_m \partial_j l_m) \right) \left( \nabla_{\boldsymbol{\theta}} \partial_p \partial_q l_m + \frac{1}{2} \nabla_{\boldsymbol{\theta}} (\partial_p l_m \partial_q l_m) \right)^\top \right] \Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^*} \\
V_{ij} &\triangleq \mathfrak{V}_{iiji} \\
W_{ij} &\triangleq \mathfrak{V}_{ijij}
\end{aligned}$$

For the proof of asymptotic normality we need the following extra assumptions.

**Assumption 9** (Lipschitz smoothness on second derivatives). *For  $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$  near  $\boldsymbol{\theta}^*$ , and  $\forall i, j$ ,*

$$\begin{aligned}
\|\nabla_{\boldsymbol{\theta}}^2 \partial_i \partial_j l_m(\mathbf{x}; \boldsymbol{\theta}_1) - \nabla_{\boldsymbol{\theta}}^2 \partial_i \partial_j l_m(\mathbf{x}; \boldsymbol{\theta}_2)\|_F &\leq M_{ij}(\mathbf{x}) \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2 \\
\|\nabla_{\boldsymbol{\theta}}^2 \partial_i l_m(\mathbf{x}; \boldsymbol{\theta}_1) \partial_j l_m(\mathbf{x}; \boldsymbol{\theta}_1) - \nabla_{\boldsymbol{\theta}}^2 \partial_i l_m(\mathbf{x}; \boldsymbol{\theta}_2) \partial_j l_m(\mathbf{x}; \boldsymbol{\theta}_2)\|_F &\leq N_{ij}(\mathbf{x}) \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2
\end{aligned}$$

and

$$\mathbb{E}_{p_d}[M_{ij}^2(\mathbf{x})] < \infty, \quad \mathbb{E}_{p_d}[N_{ij}^2(\mathbf{x})] < \infty, \quad \forall i, j.$$

**Lemma 4.** Suppose  $l_m(\mathbf{x}; \boldsymbol{\theta})$  is sufficiently smooth (Assumption 9) and  $p_{\mathbf{v}}$  has bounded moments (Assumption 2 and Assumption 8). Let  $\nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}; \mathbf{x}, \mathbf{v}_1^M) \triangleq \frac{1}{M} \sum_{i=1}^M \nabla_{\boldsymbol{\theta}}^2 \mathbf{v}_i^\top \nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v}_i + \frac{1}{2} \nabla_{\boldsymbol{\theta}}^2 (\mathbf{v}_i^\top \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))^2$ . Then  $\nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}; \mathbf{x}, \mathbf{v})$  is Lipschitz continuous, i.e., for  $\boldsymbol{\theta}_1$  and  $\boldsymbol{\theta}_2$  close to  $\boldsymbol{\theta}^*$ , there exists a Lipschitz constant  $L(\mathbf{x}, \mathbf{v}_1^M)$  such that

$$\|\nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}_1; \mathbf{x}, \mathbf{v}_1^M) - \nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}_2; \mathbf{x}, \mathbf{v}_1^M)\|_F \leq L(\mathbf{x}, \mathbf{v}_1^M) \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2,$$

and  $\mathbb{E}_{p_d, p_{\mathbf{v}}}[L^2(\mathbf{x}, \mathbf{v}_1^M)] < \infty$ .

*Proof.* First, we write out  $\nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}_1; \mathbf{x}, \mathbf{v}_1^M) - \nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}_2; \mathbf{x}, \mathbf{v}_1^M)$  according to the definitions. Let  $A_{ij}(\boldsymbol{\theta}) \triangleq \nabla_{\boldsymbol{\theta}}^2 \partial_i \partial_j l_m(\mathbf{x}; \boldsymbol{\theta})$  and  $B_{ij}(\boldsymbol{\theta}) \triangleq \nabla_{\boldsymbol{\theta}}^2 \partial_i l_m(\mathbf{x}; \boldsymbol{\theta}) \partial_j l_m(\mathbf{x}; \boldsymbol{\theta})$ . Then,

$$\nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}_1; \mathbf{x}, \mathbf{v}_1^M) - \nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}_2; \mathbf{x}, \mathbf{v}_1^M) = \frac{1}{M} \sum_{i,j,k} v_{k,i} v_{k,j} \left[ A_{ij}(\boldsymbol{\theta}_1) - A_{ij}(\boldsymbol{\theta}_2) + \frac{1}{2} (B_{ij}(\boldsymbol{\theta}_1) - B_{ij}(\boldsymbol{\theta}_2)) \right].$$

Then, Cauchy-Schwarz and Jensen's inequality give

$$\begin{aligned}
&\|\nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}_1; \mathbf{x}, \mathbf{v}_1^M) - \nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}_2; \mathbf{x}, \mathbf{v}_1^M)\|_F^2 \\
&= \sum_{l,m} \left( \frac{1}{M} \sum_{i,j,k} v_{k,i} v_{k,j} \left[ A_{ij}(\boldsymbol{\theta}_1)_{lm} - A_{ij}(\boldsymbol{\theta}_2)_{lm} + \frac{1}{2} (B_{ij}(\boldsymbol{\theta}_1)_{lm} - B_{ij}(\boldsymbol{\theta}_2)_{lm}) \right] \right)^2 \\
&\leq \sum_{l,m} \left( \frac{1}{M} \sqrt{\sum_{i,j} \left( \sum_k v_{k,i} v_{k,j} \right)^2} \cdot \sqrt{\sum_{i,j} \left[ A_{ij}(\boldsymbol{\theta}_1)_{lm} - A_{ij}(\boldsymbol{\theta}_2)_{lm} + \frac{1}{2} (B_{ij}(\boldsymbol{\theta}_1)_{lm} - B_{ij}(\boldsymbol{\theta}_2)_{lm}) \right]^2} \right)^2 \\
&\leq \sum_{l,m} \frac{1}{M^2} \left( \sum_{i,j} \left( \sum_k v_{k,i} v_{k,j} \right)^2 \right) \left( \sum_{i,j} \left( 2 \left[ A_{ij}(\boldsymbol{\theta}_1)_{lm} - A_{ij}(\boldsymbol{\theta}_2)_{lm} \right]^2 + \frac{1}{2} \left[ (B_{ij}(\boldsymbol{\theta}_1)_{lm} - B_{ij}(\boldsymbol{\theta}_2)_{lm}) \right]^2 \right) \right)
\end{aligned}$$

$$\begin{aligned}
&= \frac{1}{M^2} \left( \sum_{i,j} \left( \sum_k v_{k,i} v_{k,j} \right)^2 \right) \left( \sum_{i,j} \left( \sum_{l,m} 2 \left[ A_{ij}(\boldsymbol{\theta}_1)_{lm} - A_{ij}(\boldsymbol{\theta}_2)_{lm} \right]^2 + \sum_{l,m} \frac{1}{2} \left[ (B_{ij}(\boldsymbol{\theta}_1)_{lm} - B_{ij}(\boldsymbol{\theta}_2)_{lm})^2 \right] \right) \right) \\
&= \frac{1}{M^2} \left( \sum_{i,j,p,q} v_{p,i} v_{p,j} v_{q,i} v_{q,j} \right) \left( \sum_{i,j} \left( 2 \|A_{ij}(\boldsymbol{\theta}_1) - A_{ij}(\boldsymbol{\theta}_2)\|_F^2 + \frac{1}{2} \|V_{ij}(\boldsymbol{\theta}_1) - V_{ij}(\boldsymbol{\theta}_2)\|_F^2 \right) \right) \\
&\leq \underbrace{\frac{1}{M^2} \left( \sum_{i,j,p,q} v_{p,i} v_{p,j} v_{q,i} v_{q,j} \right)}_{\triangleq L^2(\mathbf{x}, \mathbf{v}_1^M)} \left( \sum_{i,j} \left( 2M_{ij}^2 + \frac{1}{2} N_{ij}^2 \right) \right) \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2^2
\end{aligned}$$

Next, we bound the expectation

$$\begin{aligned}
\mathbb{E}_{p_d, p_\mathbf{v}} [L^2(\mathbf{x}, \mathbf{v}_1^M)] &= \frac{1}{M^2} \mathbb{E}_{p_\mathbf{v}} \left[ \sum_{i,j,p,q} v_{p,i} v_{p,j} v_{q,i} v_{q,j} \right] \mathbb{E}_{p_d} \left[ \sum_{ij} 2M_{ij}^2(\mathbf{x}) + \frac{1}{2} N_{ij}^2(\mathbf{x}) \right] \\
&\stackrel{(i)}{\leq} O(1) \mathbb{E}_{p_\mathbf{v}} \left[ \sum_{i,j,p,q} v_{p,i} v_{p,j} v_{q,i} v_{q,j} \right] \\
&= O(1) \left( \sum_{ij} M(M-1) \mathbb{E}_{p_\mathbf{v}} [v_i v_j]^2 + M \mathbb{E}_{p_\mathbf{v}} [(v_i v_j)^2] \right) \\
&= O(1) \left( M(M-1) \|\mathbb{E}_{p_\mathbf{v}} [\mathbf{v} \mathbf{v}^\top]\|_F^2 + M \mathbb{E}_{p_\mathbf{v}} [\|\mathbf{v} \mathbf{v}^\top\|_F^2] \right) \\
&\stackrel{(ii)}{\leq} O(1) \left( M(M-1) \mathbb{E}_{p_\mathbf{v}} [\|\mathbf{v} \mathbf{v}^\top\|_F^2] + M \mathbb{E}_{p_\mathbf{v}} [\|\mathbf{v} \mathbf{v}^\top\|_F^2] \right) \stackrel{(iii)}{<} \infty,
\end{aligned}$$

where (i) is due to Assumption 9, (ii) is Jensen's, and (iii) is because of Assumption 2 and 8.  $\square$

**Lemma 5.** Assume that conditions in Theorem 1 and Lemma 1 hold, and  $p_\mathbf{v}$  has bounded higher-order moments (Assumption 8). Then

$$\text{Var}_{p_d, p_\mathbf{v}} [\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M)] = \sum_{i,j,p,q} \left[ \left( 1 - \frac{1}{M} \right) \Sigma_{ij} \Sigma_{pq} + \frac{1}{M} \mathfrak{S}_{ijpq} \right] \mathfrak{V}_{ijpq}, \quad (23)$$

where  $\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M) = \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}; \mathbf{x}, \mathbf{v}_1^M)|_{\boldsymbol{\theta}=\boldsymbol{\theta}^*}$

In particular, if  $p_\mathbf{v} \sim \mathcal{N}(0, I)$ , we have

$$\text{Var}_{p_d, p_\mathbf{v}} [\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M)] = \sum_{ij} V_{ij} + \frac{2}{M} \sum_i V_{ii} + \frac{2}{M} \sum_{i \neq j} W_{ij}.$$

If  $p_\mathbf{v}$  is the distribution of multivariate Rademacher random variables, we have

$$\text{Var}_{p_d, p_\mathbf{v}} [\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M)] = \sum_{ij} V_{ij} + \frac{2}{M} \sum_{i \neq j} W_{ij}.$$

*Proof.* Since  $\boldsymbol{\theta}^*$  is the true parameter of the data distribution, we have

$$\mathbb{E}_{p_d, p_\mathbf{v}} [\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M)] = \nabla_{\boldsymbol{\theta}} \mathbb{E}_{p_d, p_\mathbf{v}} [f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M)] = \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}^*; p_\mathbf{v}) = 0.$$

Therefore, (23) can be expanded as

$$\begin{aligned}
&\text{Var}_{p_d, p_\mathbf{v}} [\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M)] \\
&= \mathbb{E}_{p_d, p_\mathbf{v}} [\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M) \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M)^\top] \\
&= \mathbb{E} \left[ \sum_{i,j,p,q} \left( \frac{1}{M^2} \sum_{k,l} v_{k,i} v_{k,j} v_{l,p} v_{l,q} \right) \left( \nabla_{\boldsymbol{\theta}} \partial_i \partial_j l_m + \frac{1}{2} \nabla_{\boldsymbol{\theta}} (\partial_i l_m \partial_j l_m) \right) \left( \nabla_{\boldsymbol{\theta}} \partial_p \partial_q l_m + \frac{1}{2} \nabla_{\boldsymbol{\theta}} (\partial_p l_m \partial_q l_m) \right)^\top \right]
\end{aligned}$$

$$= \sum_{i,j,p,q} \underbrace{\mathbb{E}\left[\frac{1}{M^2} \sum_{k,l} v_{k,i} v_{k,j} v_{l,p} v_{l,q}\right]}_{\triangleq E_1} \underbrace{\mathbb{E}\left[\left(\nabla_{\theta} \partial_i \partial_j l_m + \frac{1}{2} \nabla_{\theta} (\partial_i l_m \partial_j l_m)\right) \left(\nabla_{\theta} \partial_p \partial_q l_m + \frac{1}{2} \nabla_{\theta} (\partial_p l_m \partial_q l_m)\right)^T\right]}_{\mathfrak{V}_{ijpq}}.$$

Continuing on  $E_1$ , we have that

$$\begin{aligned} E_1 &= \frac{1}{M^2} \sum_{k \neq l} \mathbb{E}[v_{k,i} v_{k,j}] \mathbb{E}[v_{l,p} v_{l,q}] + \frac{1}{M^2} \sum_k \mathbb{E}[v_{k,i} v_{k,j} v_{k,p} v_{k,q}] \\ &= \left(1 - \frac{1}{M}\right) \Sigma_{ij} \Sigma_{pq} + \frac{1}{M} \mathfrak{S}_{ijpq}, \end{aligned}$$

which leads to (23). Note that Assumption 2 guarantees that  $|\Sigma_{ij}| < \infty$  and Assumption 8 ensures  $|\mathfrak{S}_{ijpq}| < \infty$ .

If  $p_{\mathbf{v}} \sim \mathcal{N}(0, I)$ ,  $\mathfrak{S}$  and  $\Sigma$  have the following simpler forms

$$\begin{aligned} \Sigma_{ij} &= \delta_{ij} \\ \mathfrak{S}_{ijpq} &= \begin{cases} 3, & i = j = p = q \\ 1, & i = j \neq p = q \text{ or } i = p \neq j = q \text{ or } i = q \neq j = p \\ 0, & \text{otherwise} \end{cases}. \end{aligned}$$

Then, if we assume that the second derivatives of  $l_m$  are continuous, we have  $\partial_i \partial_j l_m = \partial_j \partial_i l_m$ , and the variance (23) can also be simplified to

$$\text{Var}_{p_d, p_{\mathbf{v}}} [\nabla_{\theta} f(\theta^*; \mathbf{x}, \mathbf{v}_1^M)] = \sum_{i \neq j} V_{ij} + \frac{M+2}{M} \sum_i V_{ii} + \frac{2}{M} \sum_{i \neq j} W_{ij} = \sum_{ij} V_{ij} + \frac{2}{M} \sum_i V_{ii} + \frac{2}{M} \sum_{i \neq j} W_{ij}.$$

Similarly, if  $p_{\mathbf{v}} \sim \mathcal{U}(\{\pm 1\}^D)$ , (23) has the simplified form

$$\text{Var}_{p_d, p_{\mathbf{v}}} [\nabla_{\theta} f(\theta^*; \mathbf{x}, \mathbf{v}_1^M)] = \sum_{ij} V_{ij} + \frac{2}{M} \sum_{i \neq j} W_{ij}.$$

□

**Theorem 3.** *With the notations and assumptions in Lemma 4, Lemma 5 and Theorem 2, we have*

$$\begin{aligned} &\sqrt{N}(\hat{\theta}_{N,M} - \theta^*) \xrightarrow{d} \\ &\mathcal{N}\left(0, \left(\nabla_{\theta}^2 J(\theta^*; p_{\mathbf{v}})\right)^{-1} \left(\sum_{i,j,p,q} \left[\left(1 - \frac{1}{M}\right) \Sigma_{ij} \Sigma_{pq} + \frac{1}{M} \mathfrak{S}_{ijpq}\right] \mathfrak{V}_{ijpq}\right) \left(\nabla_{\theta}^2 J(\theta^*; p_{\mathbf{v}})\right)^{-1}\right). \end{aligned}$$

In particular, if  $p_{\mathbf{v}} \sim \mathcal{N}(0, I)$ , then the asymptotic variance is

$$\left(\nabla_{\theta}^2 J(\theta^*)\right)^{-1} \left(\sum_{ij} V_{ij} + \frac{2}{M} \sum_i V_{ii} + \frac{2}{M} \sum_{i \neq j} W_{ij}\right) \left(\nabla_{\theta}^2 J(\theta^*)\right)^{-1}.$$

If  $p_{\mathbf{v}}$  is the distribution of multivariate Rademacher random variables, the asymptotic variance is

$$\left(\nabla_{\theta}^2 J(\theta^*)\right)^{-1} \left(\sum_{ij} V_{ij} + \frac{2}{M} \sum_{i \neq j} W_{ij}\right) \left(\nabla_{\theta}^2 J(\theta^*)\right)^{-1}.$$

*Proof.* To simplify notations, we use  $P_N h(\mathbf{x}) \triangleq \frac{1}{N} \sum_{i=1}^N h(\mathbf{x}_i, \cdot)$ , where  $h(\mathbf{x}, \cdot)$  is some arbitrary function. For example,  $\hat{J}(\theta; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM})$  can be written as  $P_N f(\theta; \mathbf{x}, \mathbf{v}_1^M)$ . By Taylor expansion, we can approximate  $P_N \nabla_{\theta} f(\hat{\theta}_{N,M}; \mathbf{x}, \mathbf{v}_1^M)$  around  $\theta^*$ :

$$0 = \nabla_{\theta} P_N f(\hat{\theta}_{N,M}; \mathbf{x}, \mathbf{v}_1^M)$$

$$= P_N \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M) + P_N \left( \nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M) + E_{\hat{\boldsymbol{\theta}}_{N,M}, \mathbf{x}, \mathbf{v}_1^M} \right) (\hat{\boldsymbol{\theta}}_{N,M} - \boldsymbol{\theta}^*), \quad (24)$$

where  $\|E_{\hat{\boldsymbol{\theta}}_{N,M}, \mathbf{x}, \mathbf{v}_1^M}\|_F \leq L(\mathbf{x}, \mathbf{v}_1^M) \|\hat{\boldsymbol{\theta}}_{N,M} - \boldsymbol{\theta}^*\|_2$  from Lemma 4 and Taylor expansion of vector-valued functions. Combining with the law of large numbers, we have

$$P_N \nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M) = \mathbb{E}_{p_d, p_v} [\nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M)] + o_p(1)$$

and

$$\left\| P_N E_{\hat{\boldsymbol{\theta}}_{N,M}} \right\|_F \leq \mathbb{E}_{p_d, p_v} [L(\mathbf{x}, \mathbf{v}_1^M)] \left\| \hat{\boldsymbol{\theta}}_{N,M} - \boldsymbol{\theta}^* \right\|_2 + o_p(1) = o_p(1) + o_p(1) = o_p(1),$$

where we used  $\mathbb{E}[L(\mathbf{x}, \mathbf{v}_1^M)] \leq \sqrt{\mathbb{E}[L^2(\mathbf{x}, \mathbf{v}_1^M)]} < \infty$  (Lemma 4) and the consistency of  $\hat{\boldsymbol{\theta}}_{N,M}$  (Theorem 2). Now returning to (24), we get

$$\begin{aligned} 0 &= P_N \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M) + \left( \mathbb{E}_{p_d, p_v} [\nabla_{\boldsymbol{\theta}}^2 f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M)] + o_p(1) \right) (\hat{\boldsymbol{\theta}}_{N,M} - \boldsymbol{\theta}^*) \\ &\Leftrightarrow \left( \nabla_{\boldsymbol{\theta}}^2 J(\boldsymbol{\theta}^*; p_v) + o_p(1) \right) \sqrt{N} (\hat{\boldsymbol{\theta}}_{N,M} - \boldsymbol{\theta}^*) = -\sqrt{N} P_N \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M). \end{aligned}$$

But of course, the central limit theorem and Lemma 5 yield

$$\begin{aligned} -\sqrt{N} P_N \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M) &\xrightarrow{d} \mathcal{N}(0, \text{Var}_{p_d, p_v} [\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*; \mathbf{x}, \mathbf{v}_1^M)]) \\ &= \mathcal{N}\left(0, \sum_{i,j,p,q} \left[ \left(1 - \frac{1}{M}\right) \Sigma_{ij} \Sigma_{pq} + \frac{1}{M} \mathfrak{S}_{ijpq} \right] \mathfrak{V}_{ijpq} \right). \end{aligned}$$

Then, Slutsky's theorem gives the desired result

$$\begin{aligned} \sqrt{N} (\hat{\boldsymbol{\theta}}_{N,M} - \boldsymbol{\theta}^*) &\xrightarrow{d} \\ \mathcal{N}\left(0, \left( \nabla_{\boldsymbol{\theta}}^2 J(\boldsymbol{\theta}^*; p_v) \right)^{-1} \left( \sum_{i,j,p,q} \left[ \left(1 - \frac{1}{M}\right) \Sigma_{ij} \Sigma_{pq} + \frac{1}{M} \mathfrak{S}_{ijpq} \right] \mathfrak{V}_{ijpq} \right) \left( \nabla_{\boldsymbol{\theta}}^2 J(\boldsymbol{\theta}^*; p_v) \right)^{-1} \right). \end{aligned}$$

In particular, if  $p_v \sim \mathcal{N}(0, I)$  or  $p_v \sim \mathcal{U}(\{\pm 1\}^D)$ , we have  $J(\boldsymbol{\theta}^*; p_v) = J(\boldsymbol{\theta}^*)$ , and therefore  $\nabla_{\boldsymbol{\theta}}^2 J(\boldsymbol{\theta}^*; p_v) = \nabla_{\boldsymbol{\theta}}^2 J(\boldsymbol{\theta}^*)$ . We can apply Lemma 5 to conclude the simplified expressions for the asymptotic variance.

□

**Corollary 1** (Consistency and asymptotic normality of score matching). *Under similar assumptions used in Theorem 2 and Theorem 3, we can also conclude that the score matching estimator  $\hat{\boldsymbol{\theta}}_N \triangleq \arg \min_{\boldsymbol{\theta} \in \Theta} \hat{J}(\boldsymbol{\theta}; \mathbf{x})$  is consistent*

$$\hat{\boldsymbol{\theta}}_N \xrightarrow{p} \boldsymbol{\theta}^*$$

and asymptotically normal

$$\sqrt{N} (\hat{\boldsymbol{\theta}}_N - \boldsymbol{\theta}^*) \xrightarrow{d} \mathcal{N}\left(0, \left( \nabla_{\boldsymbol{\theta}}^2 J(\boldsymbol{\theta}^*) \right)^{-1} \left( \sum_{ij} V_{ij} \right) \left( \nabla_{\boldsymbol{\theta}}^2 J(\boldsymbol{\theta}^*) \right)^{-1} \right).$$

*Proof.* Note that

$$\begin{aligned} \text{Var}_{p_d} [\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*; \mathbf{x})] &= \mathbb{E}_{p_d} [\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*; \mathbf{x}) \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^*; \mathbf{x})^\top] \\ &= \mathbb{E}_{p_d} \left[ \left( \sum_i \nabla_{\boldsymbol{\theta}} \partial_i \partial_i l_m + \frac{1}{2} \nabla_{\boldsymbol{\theta}} (\partial_i l_m \partial_i l_m) \right) \left( \sum_j \nabla_{\boldsymbol{\theta}} \partial_j \partial_j l_m + \frac{1}{2} \nabla_{\boldsymbol{\theta}} (\partial_j l_m \partial_j l_m) \right)^\top \right] \\ &= \sum_{ij} \mathbb{E}_{p_d} \left[ \left( \nabla_{\boldsymbol{\theta}} \partial_i \partial_i l_m + \frac{1}{2} \nabla_{\boldsymbol{\theta}} (\partial_i l_m \partial_i l_m) \right) \left( \nabla_{\boldsymbol{\theta}} \partial_j \partial_j l_m + \frac{1}{2} \nabla_{\boldsymbol{\theta}} (\partial_j l_m \partial_j l_m) \right)^\top \right] \\ &= \sum_{ij} V_{ij} \end{aligned}$$

The other part of the proof is similar to that of Theorem 2 and Theorem 3 and is thus omitted. □

## B.5 NOISE CONTRASTIVE ESTIMATION

**Proposition 1.** Define

$$J_{NCE}(\boldsymbol{\theta}) \triangleq -\mathbb{E}_{p_d}[\log h(\mathbf{x}; \boldsymbol{\theta})] - \mathbb{E}_{p_n}[\log(1 - h(\mathbf{x}; \boldsymbol{\theta}))]$$

where

$$\begin{aligned} h(\mathbf{x}; \boldsymbol{\theta}) &\triangleq \frac{p_m(\mathbf{x}; \boldsymbol{\theta})}{p_m(\mathbf{x}; \boldsymbol{\theta}) + p_m(\mathbf{x} - \mathbf{v}; \boldsymbol{\theta})} \\ p_n(\mathbf{x}) &= p_d(\mathbf{x} + \mathbf{v}). \end{aligned}$$

Then when  $\|\mathbf{v}\|_2 \rightarrow 0$ , we have

$$J_{NCE}(\boldsymbol{\theta}) = 2 \log 2 + \frac{1}{4} \mathbb{E}_{p_d} \left[ \mathbf{v}^\top \nabla^2 \log p_m(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v} + \frac{1}{2} (\nabla \log p_m(\mathbf{x}; \boldsymbol{\theta})^\top \mathbf{v})^2 \right] + o(\|\mathbf{v}\|_2^2)$$

*Proof.* Using Taylor expansion, we can immediately get

$$\log p_m(\mathbf{x} + \mathbf{v}; \boldsymbol{\theta}) = \log p_m(\mathbf{x}; \boldsymbol{\theta}) + \nabla \log p_m(\mathbf{x}; \boldsymbol{\theta})^\top \mathbf{v} + \frac{1}{2} \mathbf{v}^\top \nabla^2 \log p_m(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v} + o(\|\mathbf{v}\|_2^2).$$

Next, observe that

$$\begin{aligned} \log(p_m(\mathbf{x}; \boldsymbol{\theta}) + p_m(\mathbf{x} + \mathbf{v}; \boldsymbol{\theta})) &= \log p_m(\mathbf{x}; \boldsymbol{\theta}) + \log(1 + \exp\{\log p_m(\mathbf{x} + \mathbf{v}; \boldsymbol{\theta}) - \log p_m(\mathbf{x}; \boldsymbol{\theta})\}) \\ &= \log p_m(\mathbf{x}; \boldsymbol{\theta}) + \log \left( 1 + \exp \left\{ \nabla \log p_m(\mathbf{x}; \boldsymbol{\theta})^\top \mathbf{v} + \frac{1}{2} \mathbf{v}^\top \nabla^2 \log p_m(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v} + o(\|\mathbf{v}\|_2^2) \right\} \right) \\ &= \log p_m(\mathbf{x}; \boldsymbol{\theta}) + \log 2 + \frac{1}{2} \left[ \nabla \log p_m(\mathbf{x}; \boldsymbol{\theta})^\top \mathbf{v} + \frac{1}{2} \mathbf{v}^\top \nabla^2 \log p_m(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v} \right] + \frac{1}{8} (\nabla \log p_m(\mathbf{x}; \boldsymbol{\theta})^\top \mathbf{v})^2 + o(\|\mathbf{v}\|_2^2). \end{aligned}$$

Similarly, we have

$$\begin{aligned} \log(p_m(\mathbf{x}; \boldsymbol{\theta}) + p_m(\mathbf{x} - \mathbf{v}; \boldsymbol{\theta})) \\ = \log p_m(\mathbf{x}; \boldsymbol{\theta}) + \log 2 + \frac{1}{2} \left[ -\nabla \log p_m(\mathbf{x}; \boldsymbol{\theta})^\top \mathbf{v} + \frac{1}{2} \mathbf{v}^\top \nabla^2 \log p_m(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v} \right] + \frac{1}{8} (\nabla \log p_m(\mathbf{x}; \boldsymbol{\theta})^\top \mathbf{v})^2 + o(\|\mathbf{v}\|_2^2). \end{aligned}$$

Finally, note that

$$\begin{aligned} J_{NCE}(\boldsymbol{\theta}) &= -\mathbb{E}_{p_d}[\log h(\mathbf{x}; \boldsymbol{\theta}) + \log(1 - h(\mathbf{x} + \mathbf{v}; \boldsymbol{\theta}))] \\ &= -\mathbb{E}_{p_d}[\log p_m(\mathbf{x}; \boldsymbol{\theta}) - \log(p_m(\mathbf{x}; \boldsymbol{\theta}) + p_m(\mathbf{x} + \mathbf{v}; \boldsymbol{\theta}))] - \mathbb{E}_{p_d}[\log p_m(\mathbf{x}; \boldsymbol{\theta}) - \log(p_m(\mathbf{x}; \boldsymbol{\theta}) + p_m(\mathbf{x} - \mathbf{v}; \boldsymbol{\theta}))] \\ &= 2 \log 2 + \frac{1}{4} \mathbb{E}_{p_d} \left[ \mathbf{v}^\top \nabla^2 \log p_m(\mathbf{x}; \boldsymbol{\theta}) \mathbf{v} + \frac{1}{2} (\nabla \log p_m(\mathbf{x}; \boldsymbol{\theta})^\top \mathbf{v})^2 \right] + o(\|\mathbf{v}\|_2^2), \end{aligned}$$

as desired.  $\square$

## C ADDITIONAL DETAILS OF EXPERIMENTS

### C.1 KERNEL EXPONENTIAL FAMILIES

**Model.** The kernel exponential family is a class of densities with unnormalized log density given by  $\log \tilde{p}_f(\mathbf{x}) = f(\mathbf{x}) + \log q_0(\mathbf{x})$ . Here,  $q_0$  is a fixed function and  $f$  belongs to a reproducing kernel Hilbert space  $\mathcal{H}$ , with kernel  $k$  (Canu & Smola, 2006; Sriperumbudur et al., 2017). We see this is a member of the exponential family by using reproducing property,  $f(\mathbf{x}) = \langle f, k(\cdot, \cdot) \rangle_{\mathcal{H}}$ . Rewriting the density, the model has natural parameter  $f$  and sufficient statistic  $k(\mathbf{x}, \cdot)$ :

$$\tilde{p}_f(\mathbf{x}) = \exp(f(\mathbf{x})) q_0(\mathbf{x}) = \exp(\langle f, k(\cdot, \cdot) \rangle_{\mathcal{H}}) q_0(\mathbf{x})$$

To improve computational cost of learning  $f$ , Sutherland et al. (2018) use a Nyström-type lite approximation, selecting  $L$  inducing points  $\mathbf{z}_l$ , and  $f$  of the form:

$$f(\mathbf{x}) = \sum_{l=1}^L \alpha_l k(\mathbf{x}, \mathbf{z}_l)$$

We compare training using our objective against deep kernel exponential family (DKEF) models trained using exact score matching in Wenliang et al. (2019).

The kernel  $k(\mathbf{x}, \mathbf{y})$  is a mixture of  $R = 3$  Gaussian kernels, with features extracted by a neural network,  $\phi_{w_r}(\cdot)$ , length scales  $\sigma_r$ , and nonnegative mixture coefficients  $\rho_r$ . The neural network is a three layer fully connected network with a skip connection from the input to output layers and softplus nonlinearities. Each hidden layer has 30 neurons. We have:

$$k_w(\mathbf{x}, \mathbf{y}) = \sum_{r=1}^R \rho_r \exp\left(-\frac{1}{2\sigma_r^2} \|\phi_{w_r}(\mathbf{x}) - \phi_{w_r}(\mathbf{y})\|^2\right).$$

When training DKEF models, Wenliang et al. (2019) note that it is possible to analytically minimize the score matching loss over  $\alpha$  because the objective is quadratic in  $\alpha$ . As a result, models are trained in a two step procedure:  $\alpha$  is analytically minimized over a training minibatch, then the loss is computed over a validation minibatch. When training models, the analytically minimized  $\alpha$  is treated as function of the other parameters. By doing this, Wenliang et al. (2019) can also directly optimize the coefficient  $\lambda_\alpha$  of a  $\ell_2$  regularization loss on  $\alpha$ . This regularization coefficient is initialized to 0.01, and is trained. (More details about the two-step optimization procedure can be found in Wenliang et al. (2019), which also includes a finalization stage for  $\alpha$ ).

A similar closed form for sliced score matching, denoising score matching, approximate backpropogation, and curvature propagation can be derived. The derivation for sliced score matching is presented below for completeness.

**Proposition 2.** Consider the loss

$$\hat{J}(\boldsymbol{\theta}, \lambda_\alpha; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) = \hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) + \frac{1}{2} \lambda_\alpha \|\boldsymbol{\alpha}\|_2^2$$

where

$$\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) = \frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M \left[ \mathbf{v}_{ij}^\top \nabla^2 \log p_m(\mathbf{x}_i) \mathbf{v}_{ij} + \frac{1}{2} (\mathbf{v}_{ij}^\top \nabla \log p_m(\mathbf{x}_i))^2 \right].$$

For fixed  $k$ ,  $\mathbf{z}$ , and  $\lambda_\alpha$ , as long as  $\lambda_\alpha > 0$  then the optimal  $\boldsymbol{\alpha}$  is

$$\begin{aligned} \boldsymbol{\alpha}(\lambda_\alpha, k, \mathbf{z}, \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) &= \arg \min_{\boldsymbol{\alpha}} \hat{J}(\boldsymbol{\theta}, \lambda_\alpha; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) = -(\mathbf{G} + \lambda_\alpha \mathbf{I})^{-1} \mathbf{b} \\ G_{l,l'} &= \frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M (\mathbf{v}_{ij}^\top \nabla k(\mathbf{x}_i, \mathbf{z}_l)) (\mathbf{v}_{ij}^\top \nabla k(\mathbf{x}_i, \mathbf{z}_{l'})) \\ b_l &= \frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M \mathbf{v}_{ij}^\top \nabla^2 k(\mathbf{x}_i, \mathbf{z}_l) \mathbf{v}_{ij} + (\mathbf{v}_{ij}^\top \nabla \log q_0(\mathbf{x}_i)) (\mathbf{v}_{ij}^\top \nabla k(\mathbf{x}_i, \mathbf{z}_l)). \end{aligned}$$

*Proof.* The derivation follows very similarly to Proposition 3 in Wenliang et al. (2019). We will show that the loss is quadratic in  $\boldsymbol{\alpha}$ . Note that

$$\begin{aligned} \frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M \mathbf{v}_{ij}^\top \nabla^2 \log p_m(\mathbf{x}_i) \mathbf{v}_{ij} &= \frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M \left[ \sum_{m=1}^M \alpha_l \mathbf{v}_{ij}^\top \nabla^2 k(\mathbf{x}_i, \mathbf{z}_l) \mathbf{v}_{ij} \right] + C \\ &= \boldsymbol{\alpha}^\top \left[ \frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M \mathbf{v}_{ij}^\top \nabla^2 k(\mathbf{x}_i, \mathbf{z}_l) \mathbf{v}_{ij} \right]_l + C \end{aligned}$$

$$\begin{aligned}
\frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M \frac{1}{2} (\mathbf{v}_{ij}^\top \nabla \log p_m(\mathbf{x}_i))^2 &= \frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M \frac{1}{2} \left( \sum_{m,m'=1}^M \alpha_l \alpha_{m'} (\mathbf{v}_{ij}^\top \nabla k(\mathbf{x}_i, \mathbf{z}_l)) (\mathbf{v}_{ij}^\top \nabla k(\mathbf{x}_i, \mathbf{z}_{l'})) \right. \\
&\quad \left. + 2 \sum_{m=1}^M \alpha_l (\mathbf{v}_{ij}^\top \nabla \log q_0(\mathbf{x}_i)) (\mathbf{v}_{ij}^\top \nabla k(\mathbf{x}_i, \mathbf{z}_l)) + (\mathbf{v}_{ij}^\top \nabla \log q_0(\mathbf{x}_i))^2 \right) \\
&= \frac{1}{2} \boldsymbol{\alpha}^\top \mathbf{G} \boldsymbol{\alpha} + \boldsymbol{\alpha}^\top \left[ \frac{1}{N} \frac{1}{M} \sum_{i=1}^N \sum_{j=1}^M (\mathbf{v}_{ij}^\top \nabla \log q_0(\mathbf{x}_i)) (\mathbf{v}_{ij}^\top \nabla k(\mathbf{x}_i, \mathbf{z}_l)) \right] + C.
\end{aligned}$$

Thus the overall optimization problem is

$$\begin{aligned}
\boldsymbol{\alpha}(\lambda_\alpha, k, \mathbf{z}, \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) &= \arg \min_{\boldsymbol{\alpha}} \hat{J}(\boldsymbol{\theta}, \lambda_\alpha; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) \\
&= \arg \min_{\boldsymbol{\alpha}} \frac{1}{2} \boldsymbol{\alpha}^\top (\mathbf{G} + \lambda_\alpha \mathbf{I}) \boldsymbol{\alpha} + \boldsymbol{\alpha}^\top \mathbf{b}.
\end{aligned}$$

Because  $\lambda_\alpha > 0$  and  $\mathbf{G}$  is positive semidefinite, the matrix in parentheses is strictly positive definite, and the claimed result follows directly from standard vector calculus.  $\square$

**Hyperparameters.** RedWine and WhiteWine are dequantized by adding uniform noise to each dimension in the range  $[-d, d]$  where  $d$  is the median distance between two values for that dimension. For each dataset, 10% of the entire data was used as testing, and 10% of the remaining was used for validation. PCA whitening is applied to the data. Noise of standard deviation 0.05 is added as part of preprocessing.

The DKEF models have  $R = 3$  Gaussian kernels. Each feature extractor is a 3-layer neural network with a skip connection from the input to output, with 30 hidden neurons per layer. Weights were initialized from a Gaussian distribution with standard deviation equal to  $\frac{1}{\sqrt{30}}$ . Length scales  $\sigma_r$  were initialized to 1.0, 3.3 and 10.0. We use  $L = 200$  trainable inducing points, which were initialized from training data.

Models are trained using an Adam optimizer, with learning rate  $10^{-2}$ . A batch size of 200 is used, with 100 points for computing  $\boldsymbol{\alpha}$ , and 100 for computing the loss. Models are stopped after validation loss does not improve for 200 steps.

For denoising score matching, we perform a grid search with values [0.02, 0.04, 0.06, 0.08, 0.10, 0.12, 0.14, 0.16, 0.20, 0.24, 0.28, 0.32, 0.40, 0.48, 0.56, 0.64, 1.28]. We train models for each value of  $\sigma$  using two random seeds, and pick the  $\sigma$  with the best average validation score matching loss. For curvature propagation, one noise sample is used to match the performance of sliced score matching.

**Log-likelihoods.** Log-likelihoods are presented below. They are estimated using AIS, using a proposal distribution  $\mathcal{N}(0, 2I)$ , using 1,000,000 samples.

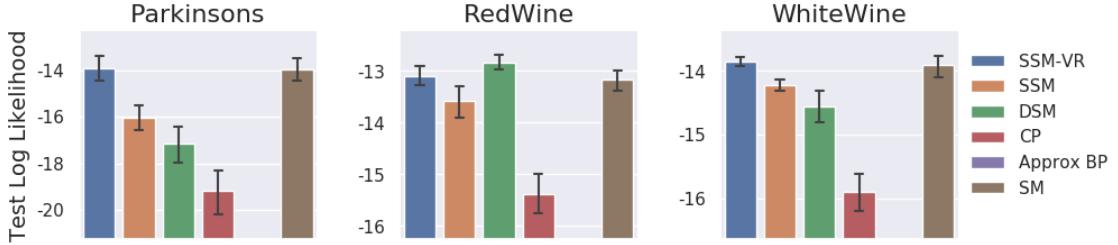


Figure 3: Log-likelihoods after training DKEF models on UCI datasets with different loss functions; higher is better. Results for approximate backpropagation are not shown because log-likelihoods were smaller than  $-10^6$ .

Name	Configuration	Algorithm
Encoder	Linear(784, 256), Tanh Linear(256, 256), Tanh Linear(256, $D_z$ )	ELBO VAE, WAE
Implicit Encoder	Linear( $784 + D_\epsilon$ , 256), Tanh Linear(256, 256), Tanh Linear(256, $D_z$ )	Implicit VAE
Decoder	Linear( $D_z$ , 256), Tanh Linear(256, 256), Tanh Linear(256, 784), Sigmoid	All
Score Estimator	Linear( $784 + D_z$ , 256), Tanh Linear(256, 256), Tanh Linear(256, $D_z$ )	Implicit VAE
Score Estimator	Linear( $D_z$ , 256), Tanh Linear(256, 256), Tanh Linear(256, $D_z$ )	WAE

Table 8: Architectures on MNIST. In our models,  $D_\epsilon = D_z$ .  $D_z$  takes the values 8 and 32 in different experiments.

## C.2 NICE

**Hyperparameters and Model Architecture.** The model has four coupling layers, each with five hidden layers, for a total of 20 hidden layers, as well as a final scale layer (Dinh et al., 2015). Softplus nonlinearities are used between hidden layers.

Models are trained using the Adam optimizer with learning rate  $10^{-3}$  for 100 epochs. The best checkpoint on exact score matching loss, evaluation every 100 iterations, is used to report test set performance. We use a batch size of 128.

Data are dequantized by adding uniform noise in the range  $[-\frac{1}{512}, \frac{1}{512}]$ , clipped to be in the range  $[-0.001, 0.001]$ , and then transformed using a logit transformation  $\log(x) - \log(1 - x)$ . 90% of the training set is used for training, and 10% for validation, and the standard test set is used.

For grid search for the optimal value of  $\sigma$ , eight values are used: [0.01, 0.05, 0.10, 0.20, 0.28, 0.50, 1.00, 1.50]. We also evaluate  $\sigma = 1.74$ , chosen by the heuristic in Saremi et al. (2018). The model with the best performance on validation score matching loss is used. Only nine values of  $\sigma$  are evaluated because training each model takes approximately two hours.

## C.3 SCORE ESTIMATION FOR IMPLICIT DISTRIBUTIONS

**Architectures.** We put the architectures of all networks used in the MNIST and CelebA experiments in Tab. 8 and Tab. 9 respectively.

**Training.** For MNIST experiments, we use RMSProp optimizer with a learning rate of 0.001 for all methods. On CelebA, the learning rate is changed to 0.0001. All algorithms are trained for 100000 iterations with a batch size of 128.

**Samples.** All samples are generated after 100000 training iterations.

## D VARIANCE REDUCTION

Below we discuss approaches to reduce the variance of  $\hat{J}(\theta; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM})$ , which can lead to better performance in practice. The most naive approach, of course, is using a larger  $M$  to compute  $\hat{J}(\theta; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM})$ . However, this requires more computation and when  $M$  is close to the data dimension, sliced score matching will lose its computational advantage over score matching.

An alternative approach is to leverage control variates (Owen, 2013). A control variate is a random variable whose expectation is tractable, and is highly correlated with another random variable without a tractable expectation. Define

Name	Configuration	Algorithm
Encoder	$5 \times 5$ conv; $m$ maps; stride $2 \times 2$ ; padding 2, ReLU $5 \times 5$ conv; $2m$ maps; stride $2 \times 2$ ; padding 2, ReLU $5 \times 5$ conv; $4m$ maps; stride $2 \times 2$ ; padding 2, ReLU $5 \times 5$ conv; $8m$ maps; stride $2 \times 2$ ; padding 2, ReLU 512 Dense, ReLU $D_z$ Dense	ELBO VAE, WAE
Implicit Encoder	concat [ $\mathbf{x}$ , ReLU(Dense( $\epsilon$ ))] along channels $5 \times 5$ conv; $m$ maps; stride $2 \times 2$ ; padding 2, ReLU $5 \times 5$ conv; $2m$ maps; stride $2 \times 2$ ; padding 2, ReLU $5 \times 5$ conv; $4m$ maps; stride $2 \times 2$ ; padding 2, ReLU $5 \times 5$ conv; $8m$ maps; stride $2 \times 2$ ; padding 2, ReLU 512 Dense, ReLU $D_z$ Dense	Implicit VAE
Decoder	Dense, ReLU $5 \times 5$ conv $^\top$ ; $4m$ maps; stride $2 \times 2$ ; padding 2; out padding 1, ReLU $5 \times 5$ conv $^\top$ ; $2m$ maps; stride $2 \times 2$ ; padding 2; out padding 1, ReLU $5 \times 5$ conv $^\top$ ; $1m$ maps; stride $2 \times 2$ ; padding 2; out padding 1, ReLU $5 \times 5$ conv $^\top$ ; $c$ maps; stride $2 \times 2$ ; padding 2; out padding 1, Tanh	All
Score Estimator	concat [ $\mathbf{x}$ , ReLU(Dense( $\mathbf{z}$ ))] along channels $5 \times 5$ conv; $m$ maps; stride $2 \times 2$ ; padding 2, ReLU $5 \times 5$ conv; $2m$ maps; stride $2 \times 2$ ; padding 2, ReLU $5 \times 5$ conv; $4m$ maps; stride $2 \times 2$ ; padding 2, ReLU $5 \times 5$ conv; $8m$ maps; stride $2 \times 2$ ; padding 2, ReLU 512 Dense, ReLU $D_z$ Dense	Implicit VAE
Score Estimator	Reshape(ReLU(Dense( $\mathbf{z}$ ))) to 1 channel $5 \times 5$ conv; $m$ maps; stride $2 \times 2$ ; padding 2, ReLU $5 \times 5$ conv; $2m$ maps; stride $2 \times 2$ ; padding 2, ReLU $5 \times 5$ conv; $4m$ maps; stride $2 \times 2$ ; padding 2, ReLU $5 \times 5$ conv; $8m$ maps; stride $2 \times 2$ ; padding 2, ReLU 512 Dense, ReLU $D_z$ Dense	WAE

Table 9: Architectures on CelebA. In our models,  $D_\epsilon = D_z$ .  $D_z$  takes the values 8 or 32 in different experiments.

$c(\boldsymbol{\theta}; \mathbf{x}, \mathbf{v}) \triangleq \frac{1}{2}(\mathbf{v}^\top \mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta}))^2$ . Note that when  $p_{\mathbf{v}}$  is a multivariate standard normal or multivariate Rademacher distribution,  $c(\boldsymbol{\theta}; \mathbf{x}, \mathbf{v})$  will have a tractable expectation, *i.e.*,

$$\mathbb{E}_{p_{\mathbf{v}}}[c(\boldsymbol{\theta}; \mathbf{x}, \mathbf{v})] = \frac{1}{2} \|\mathbf{s}_m(\mathbf{x}; \boldsymbol{\theta})\|_2^2,$$

which is easily computable. Now let  $\beta(\mathbf{x})c(\boldsymbol{\theta}; \mathbf{x}, \mathbf{v})$  be our control variate, where  $\beta(\mathbf{x})$  is a function to be determined. Due to the structural similarity between  $\beta(\mathbf{x})c(\boldsymbol{\theta}; \mathbf{x}, \mathbf{v})$  and  $\hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM})$ , it is easy to believe that  $\beta(\mathbf{x})c(\boldsymbol{\theta}; \mathbf{x}, \mathbf{v})$  can be a correlated control variate with an appropriate  $\beta(\mathbf{x})$ . We thus consider the following objective

$$\hat{J}_{\text{vr}}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) \triangleq \hat{J}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM}) - \frac{1}{N} \sum_{i=1}^N \beta(\mathbf{x}_i) \left( \frac{1}{M} \sum_{j=1}^M c(\boldsymbol{\theta}; \mathbf{x}_i, \mathbf{v}_{ij}) - \frac{1}{2} \|\mathbf{s}_m(\mathbf{x}_i; \boldsymbol{\theta})\|_2^2 \right).$$

Note that  $\mathbb{E}[\hat{J}_{\text{vr}}(\boldsymbol{\theta}; \mathbf{x}_1^N, \mathbf{v}_{11}^{NM})] = J(\boldsymbol{\theta}; p_{\mathbf{v}})$ . The theory of control variates guarantees the existence of  $\beta(\mathbf{x})$  that can reduce the variance. In practice, there can be many heuristics of choosing  $\beta(\mathbf{x})$ , and we found that  $\beta(\mathbf{x}) \equiv 1$  can often be a good choice in our experiments.

## E PSEUDOCODE

---

**Algorithm 2** Score Matching

---

**Input:**  $\tilde{p}_m(\cdot; \theta), \mathbf{x}$

```
1:  $\mathbf{s}_m(\mathbf{x}; \theta) \leftarrow \text{grad}(\log \tilde{p}_m(\mathbf{x}; \theta), \mathbf{x})$ 
2:  $J \leftarrow \frac{1}{2} \|\mathbf{s}_m(\mathbf{x}; \theta)\|_2^2$ 
3: for  $d \leftarrow 1$  to  $D$  do ▷ For each diagonal entry
4:    $(\nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \theta))_d \leftarrow \text{grad}((\mathbf{s}_m(\mathbf{x}; \theta))_d, \mathbf{x})_d$ 
5:    $J \leftarrow J + (\nabla_{\mathbf{x}} \mathbf{s}_m(\mathbf{x}; \theta))_d$ 
6: end for
return  $J$ 
```

---