Exponential Family Estimation via Dynamics Embedding

*Bo Dai[1], *Hanjun Dai[2], Niao He[3], Arthur Gretton[4], Le Song[2], Dale Schuurmans[1]
[1]Google Brain, [2]Georgia Tech, [3] UIUC, [4] UCL

Abstract

We present an efficient algorithm to solve maximum likelihood estimation (MLE) for general exponential family, even in cases when the natural parameter is represented by a deep neural network. Popular algorithms such as Contrastive Divergence require computing an expensive sampler through Markov chain Monte Carlo methods and are inefficient. Rooted in a novel primal-dual view of the MLE, the proposed algorithm jointly learns the model as well as a dual sampler for the exponential family. Inspired by the recent success in dynamics-based sampling algorithms, we further parametrize the sampler by mimicking various dynamical systems, resulting in different neural network architectures as *dynamics embeddings*. Through coupled model and sampler learning, the proposed algorithm achieves both efficient sampling and estimation. Finally, we test the proposed estimator empirically on several synthetic benchmarks.

1 Introduction

The exponential family is one of the most important classes of distributions in statistics and machine learning. It includes many commonly used distributions, *e.g.*, Gaussian, Poisson and multinomial distribution, and undirected graphical models, *e.g.*, restricted Boltzmann machine and Markov random field. The natural form of the exponential family over $\Omega \subset \mathbb{R}^d$ with the sufficient statistics $f(\cdot) : \Omega \to \mathbb{R}$ is defined as

$$p_f(x) = \exp\left(f(x) - A(f)\right), \tag{1}$$

where $A(f) := \log \int_\Omega \exp\left(f(x)\right) dx$ and $f \in \mathcal{F}$ denotes the valid parametrization family. Given samples $\mathcal{D} = [x_i]_{i=1}^N$, the maximum log-likelihood estimation (MLE) of the exponential family, *i.e.*,

$$\max_{f \in \mathcal{F}} \; L(f) = \widehat{\mathbb{E}}_{\mathcal{D}}\left[f(x)\right] - A(f), \tag{2}$$

has already been well-studied, whose desirable statistical properties such as asymptotic unbiasedness, consistency and asymptotic normality have been established. However, due to the intractability of $A(f)$, the MLE optimization (2) is difficult to solve. Contrastive Divergence (CD) [Hinton, 2002] has been one of the dominant methods for exponential family estimation. Observing that the gradient $\frac{\partial L(f)}{\partial f} = \mathbb{E}_{\mathcal{D}}\left[\nabla_f f(x)\right] - \mathbb{E}_{p_f(x)}\left[\nabla_f f(x)\right]$, the CD algorithm applies stochastic gradient descent with MCMC samples. When the MCMC steps $T \to \infty$, the bias of the stochastic gradient approaches 0, yet the gradient computation becomes prohibitive. On the other hand, when T is small, computing the stochastic gradient is efficient, but the bias could be significant, leading to suboptimal solutions. In summary, how to efficiently solve the MLE optimization has not been answered properly.

In this paper, we exploit the *primal-dual* view of the MLE, which bypasses the computation of the intractable $\log$-partition function in Section 2.1. Moreover, such a novel view provides a natural objective for jointly learning a sampler for the model, as an alternative to the expensive MCMC in the CD algorithm and inference stage. Inspired by the dynamics-based sampling algorithms, *e.g.*,

*indicates equal contributions.

Third workshop on Bayesian Deep Learning (NeurIPS 2018), Montréal, Canada.

Hamiltonian Monte-Carlo (HMC) [Neal et al., 2011], we introduce a new technique, called *dynamics embedding*, to parametrize the sampler in Section 2.2, that exhibits both flexibility and computation efficiency. In Section 2.3, we provide a novel algorithm for exponential family estimation, which learns the model and the sampler jointly through back-propagate. Finally, we test the proposed algorithm and demonstrate its superior performance on several synthetic benchmarks in Section 3.

2 MLE Estimation via Learning to Sample

In this section, we introduce two key components of the framework, *i.e.*, the *primal-dual view* of the MLE and *dynamics embedding* sampler, which play vital roles in deriving the novel algorithm.

2.1 The Primal-Dual View of MLE

First, observe that the log-partition function $A(f)$ can be rewritten as follows via Fenchel duality.

Theorem 1 (Fenchel duality of log-partition) *Let $H(q) := -\int_\Omega q(x) \log q(x) dx$, we have:*

$$A(f) = \max_{q \in \mathcal{P}} \ \langle q(x), f(x) \rangle + H(q), \tag{3}$$

$$p_f(x) = \operatorname*{argmax}_{q \in \mathcal{P}} \ \langle q(x), f(x) \rangle + H(q), \tag{4}$$

where $\mathcal{P}$ denotes the space of distributions, $\langle f, g \rangle = \int_\Omega f(x) g(x) dx$.

Therefore, plugging the Fenchel dual of $A(f)$ into the MLE, we achieve a saddle-point optimization:

$$\max_{f \in \mathcal{F}} \ L(f) \Leftrightarrow \max_{f \in \mathcal{F}} \min_{q \in \mathcal{P}} \ \ell(f, q) := \widehat{\mathbb{E}}_{\mathcal{D}}[f(x)] - \mathbb{E}_{q(x)}[f(x)] - H(q). \tag{5}$$

The saddle-point reformulation of MLE in (5) bypasses the explicit computation of the partition function, which is often intractable. To solve the saddle-point problem, it is essential to introduce an appropriate parametrization to the dual distribution $q(x)$. In the next section, we will introduce a new parametrization technique, called "dynamics embedding", for parametrizing the dual distribution. Before that, we first discuss the relationships between the primal-dual view and existing algorithms.

Remark (connection to Contrastive Divergence): By Theorem 1, the optimal solution to the inner optimization is $p_f(x)$. By Danskin's theorem [Bertsekas, 1995], the gradient w.r.t. $L(f)$ is exactly $\frac{\partial L(f)}{\partial f} = \widehat{\mathbb{E}}_{\mathcal{D}}[\nabla_f f(x)] - \mathbb{E}_{p_f(x)}[\nabla_f f(x)]$. To estimate the integral $\mathbb{E}_{p_f}[\nabla_f f(x)]$, the CD algorithm uses MCMC samples from $p_f(x)$ to estimate the gradient, which is computationally expensive. In practice, one may use finite step MCMC to approximate $p_f(x)$, but this will induce large bias. Instead, the primal-dual view allows us to learn a sampler rather than using MCMC.

Remark (connection to energy-based GAN): When $\mathcal{F}$ is convex and compact, then *strong duality* holds, *i.e.*, $\max_{f \in \mathcal{F}} \min_{q \in \mathcal{P}} \ \ell(f, q) = \min_{q \in \mathcal{P}} \max_{f \in \mathcal{F}} \ \ell(f, q)$. In particular, if $\mathcal{F}$ is chosen to be the set of all 1-Lipschitz functions, the saddle point problem in (5) is equivalent to WGAN [Dai et al., 2017] with entropy regularization. This implies that the entropy-regularized WGAN provides an estimation to the exponential family. Although [Dai et al., 2017] also exploits such a dual representation, their algorithm requires either a heuristic approximation or a lower bound of the entropy, due to their naive parametrization of the $q(x)$. To avoid the entropy term, [Kim and Bengio, 2016] estimates q by minimizing reverse KL-divergence and parametrizing the distribution via a deep directed graphical model, whose approximation ability is limited to known distributions [Kingma and Welling, 2013, Mnih and Gregor, 2014]. Hence, both of these two algorithms suffer from large approximation errors and may lead to sub-optimal solutions.

2.2 Dynamics Embedding for Sampler Parametrization

A wide spectrum of techniques for distribution parametrization are developed in the context of variational inference and generative models, such as the reparametrization trick [Kingma and Welling, 2013, Rezende et al., 2014], transport mapping [Goodfellow et al., 2014], dynamic flows [Rezende and Mohamed, 2015, Kingma et al., 2016, Tomczak and Welling, 2016, Dinh et al., 2016], optimization embedding [Dai et al., 2018], and etc. However, for exponential family estimation, careful parametrization of dual distribution is critical in two-fold: **i)** the parametrization family needs to be flexible enough to diminish the error from solving the inner minimization problem; **ii)** the parametrized representation should be able to provide density values in order to compute the entropy function $H(q)$ in (2). Neither the reparametrization trick nor the transport mapping would

Algorithm 1 MLE via Learning to Sample

1: Initialize $\Theta_1 = \left(\theta_1, \eta_1, M_1^{-1}\right)$ randomly, set length of steps T.
2: **for** iteration $k = 1, \ldots, K$ **do**
3: Sample mini-batch $\{x_i\}_{i=1}^m$ from dataset $\mathcal{D}$ and $\left\{x_i^0, v_i^0\right\}_{i=1}^m$ from $q_\theta^0(x, v)$.
4: **for** iteration $t = 1, \ldots, T$ **do**
5: Compute $(x^t, v^t) = \mathbf{L}_{f,M}\left(x^{t-1}, v^{t-1}\right)$ for each pair of $\left\{x_i^0, v_i^0\right\}_{i=1}^m$.
6: **end for**
7: **[Learning the sampler]** $\Theta_{k+1} = \Theta_k - \gamma_k \hat{\nabla}_\Theta \ell\left(f_k; \Theta_k\right)$
8: **[Estimating the exponential family]** $f_{k+1} = f_k - \gamma_k \hat{\nabla}_f \ell\left(f_k; \Theta_k\right)$.
9: **end for**

be sufficient here. To achieve both aspects, we propose to treat the dual distributions with flow-based parametrization.

Inspired by the dynamic-based sampling algorithms, we propose dynamic embedding for parametrizing the dual distributions that achieves both aspects. To begin with, we employ the Hamiltonian dynamics [Caterini et al., 2018] for dual distribution parametrization. This technique can be extended to other dynamics, *e.g.*, generalized Hamiltonian dynamics, deterministic Langevin dynamics, and continuous-time Langevin dynamics, each with its own advantages. We provide these extensions in Appendix B.1, B.2, B.3, respectively.

Define the Hamiltonian function as $\mathcal{H}(x, v) = -f(x) + k(v)$, where $k(v) = \frac{1}{2} v^\top M v$ is the kinetic energy. The Hamiltonian dynamics generate (x, v) over time t, following $\left[\frac{dx}{dt}, \frac{dv}{dt}\right] = \left[\partial_v \mathcal{H}(x, v), -\partial_x \mathcal{H}(x, v)\right] = \left[M^{-1} v, \nabla_x f(x)\right]$. The vanilla HMC is derived by discretizing the Hamiltonian dynamics with leapfrog integrator, with a single time step:

$$v^{\frac{1}{2}} = v + \frac{\eta}{2} \nabla_x f(x); \quad x' = x + \eta M^{-1} v^{\frac{1}{2}}; \quad v' = v + \frac{\eta}{2} \nabla_x f(x), \tag{6}$$

where η is defined as the leapfrog stepsize. Denote the one-step leapfrog as $(x', v') = \mathbf{L}_{f,M}(x, v)$ and assume the $\left(x^0, v^0\right) \sim q_\theta^0(x)\, p(v)$. After T iterations, we obtain $\left(x^T, v^T\right) = \mathbf{L}_{f,M} \circ \mathbf{L}_{f,M} \circ \ldots \circ \mathbf{L}_{f,M}\left(x^0, v^0\right)$ as a neural network, which we term the *dynamics embedding*. It can be verified that the transformations from (x, v) to $\left(x, v^{-\frac{1}{2}}\right)$ and from $\left(x, v^{-\frac{1}{2}}\right)$ to (x', v') are all shear mappings whose determinant of the Jacobian is 1. Therefore, we have $q_{f,M}^T\left(x^T, v^T\right) = q_\theta^0\left(x^0, v^0\right)$, with no need for calculating the determinant of the Jacobian.

2.3 Coupled Model and Sampler Learning

Combing the saddle-point reformulation (5) with the dynamics embedding parametrization for the dual distribution, we arrive at the objective

$$\max_{f \in \mathcal{F}} \min_{\Theta := [\theta, M^{-1}, \eta]} \ell(f; \Theta) := \widehat{\mathbb{E}}_{\mathcal{D}}[f] - \mathbb{E}_{(x^0, v^0) \sim q_\theta^0(x, v)}\left[f\left(x^T\right) - \log q_\theta^0(x, v)\right]. \tag{7}$$

We can now use stochastic gradient descent to jointly estimate f for the exponential families as well as the initialization q_0 and the HMC updates M and η. Note that the generated sample $\left(x_f^T, v_f^T\right)$ is a function of f. Thus the gradient w.r.t. f should take these variables into account as back-propagation through time (BPTT), *i.e.*,

$$\nabla_f \ell(f; \Theta) = \widehat{\mathbb{E}}_{\mathcal{D}}\left[\nabla_f f(x)\right] - \mathbb{E}_{(x^0, v^0) \sim q_\theta^0(x, v)}\left[\nabla_f f\left(x^T\right) + \nabla_x f\left(x^T\right) \nabla_f x^T\right]. \tag{8}$$

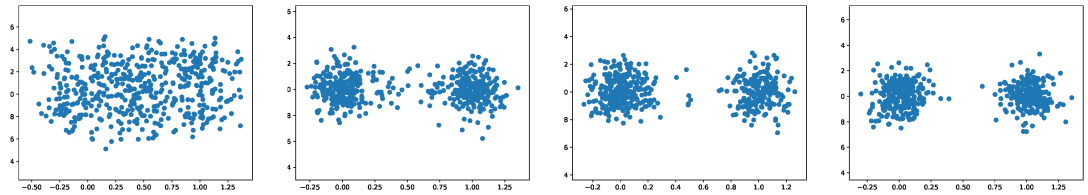
We illustrate the algorithm with Hamiltonian embedding in Algorithm 1. Extensions to other dynamics embedding are provided in Appendix B, where they differ only in $\mathbf{L}$ in Step 5.

Remark (Generalized Hamiltonian Embedding): [Levy et al., 2017] generalizes the leapfrog for better flexibility by introducing extra learnable components and combining the variable partition architecture [Dinh et al., 2016], which leads to a new embedded neural network layer $\mathbf{L}$. The determinant of the Jacobian matrix for the new $\mathbf{L}$ is also easy to compute. We can apply the modified $\mathbf{L}$ to neural network and integrate with (7) to learn the sampler as well as the exponential family. As discussed earlier, solving the inner minimization provides a sampler for $p_f(x)$. Comparing to the algorithm in Levy et al. [2017], our sampler avoids the expensive acceptance-rejection steps, yet approaches to the target distribution. More importantly, the model learning is coupled with the sampler learning in a unified framework as we explained. The details are provided in Appendix B.1.

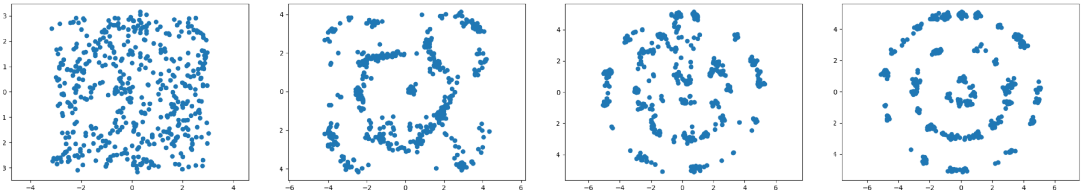
3 Experiments

In this section, we justify the proposed algorithm empirically with some preliminary experiments. We test the proposed algorithm with Hamiltonian dynamics embedding (HDE) on the `rings` and `grid` synthetic benchmarks in Sutherland et al. [2017]. The `ring` dataset contains the points uniformly sampled along three circles with radii $(1, 3, 5) \in \mathbb{R}^2$ and $\mathcal{N}\left(0, 0.1^2\right)$ noise in the radial direction and extra dimensions. The 2-dim `grid` dataset contains samples from mixture of 2 Gaussians. As we can see, both of these datasets contains the density with gaps between the modes. We parametrize f by RBF network, *i.e.*, $f(x) = \sum_{i=1}^{k} \alpha_i \exp\left(-\|x_i - x\|_2^2 / \sigma^2\right)$, both the weights $\{\alpha_i\}_{i=1}^{k}$ and the centers $\{x_i\}_{i=1}^{k}$ are learned. As the preliminary experiment, we set the $T = 5$ and the initial distribution is set to be mixture of Gaussians with $k = 500$.

We visualize the samples generated from the learned sampler iteratively on `grid` and `rings` datasets in and Figure 1, respectively. We initialize with a random dual distribution. As the algorithm iterates, the distribution converges to the target true distribution. The results demonstrate the convergence of the HDE and shows the ability of the HDE for estimating the exponential family on complicated data.



(a) initialization (b) 200-th (c) 400-th (d) 600-th



(e) initialization (f) 500-th (g) 1600-th (h) 2100-th

Figure 1: The HDE estimators on `grid`, in the first row, and `rings`, in the second row, in each iteration. The blue points are sampled from the learned model. With the algorithm proceeds, the learned distribution converges to the ground-truth.

4 Conclusion

We are targeting on the MLE estimation of the exponential family. To bypass the intractable partition function, we exploit the *primal-dual* view of the MLE, which allows us to learn a sampler explicitly to replace the expensive MCMC sampling used in CD algorithm. Inspired by the success in dynamics-based sampling algorithms, we further propose the neural network parametrization for the sampler to mimic various dynamical systems. We demonstrate the proposed algorithm is promising on several synthetic datasets. In future, we will conduct comprehensive comparison with the existing exponential family estimation methods.

References

D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 1995.

Christos Boutsidis, Petros Drineas, Prabhanjan Kambadur, Eugenia-Maria Kontopoulou, and Anastasios Zouzias. A randomized algorithm for approximating the log determinant of a symmetric positive definite matrix. *Linear Algebra and its Applications*, 533:95–117, 2017.

Anthony L Caterini, Arnaud Doucet, and Dino Sejdinovic. Hamiltonian variational auto-encoder. *arXiv preprint arXiv:1805.11328*, 2018.

Bo Dai, Hanjun Dai, Niao He, Weiyang Liu, Zhen Liu, Jianshu Chen, Lin Xiao Xiao, and Le Song. Coupled variational bayes via optimization embedding. In *NIPS*, 2018.

Zihang Dai, Amjad Almahairi, Philip Bachman, Eduard Hovy, and Aaron Courville. Calibrating energy-based generative adversarial networks. *arXiv preprint arXiv:1702.01691*, 2017.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

Will Grathwohl, Ricky TQ Chen, Jesse Betterncourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

Insu Han, Dmitry Malioutov, and Jinwoo Shin. Large-scale log-determinant computation through stochastic chebyshev expansions. In *International Conference on Machine Learning*, pages 908–917, 2015.

Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

Taesup Kim and Yoshua Bengio. Deep directed generative models with energy-based probability estimation. *arXiv preprint arXiv:1606.03439*, 2016.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Diederik P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, pages 4743–4751, 2016.

Daniel Levy, Matthew D Hoffman, and Jascha Sohl-Dickstein. Generalizing hamiltonian monte carlo with neural networks. *arXiv preprint arXiv:1711.09268*, 2017.

Andriy Mnih and Karol Gregor. Neural variational inference and learning in belief networks. *arXiv preprint arXiv:1402.0030*, 2014.

Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2(11), 2011.

Danilo J Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1278–1286, 2014.

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

Dougal J Sutherland, Heiko Strathmann, Michael Arbel, and Arthur Gretton. Efficient and principled score estimation with nystr\" om kernel exponential families. *arXiv preprint arXiv:1705.08360*, 2017.

Jakub M Tomczak and Max Welling. Improving variational auto-encoders using householder flow. *arXiv preprint arXiv:1611.09630*, 2016.

Linfeng Zhang, Weinan E, and Lei Wang. Monge-amp\ere flow for generative modeling. *arXiv preprint arXiv:1809.10188*, 2018.

Appendix

A Proof of Theorem 1

Proof Denote $l(q) := \langle q(x), f(x)\rangle + H(q)$, which is strongly concave w.r.t. $q \in \mathcal{P}$, the optimal q^* can be obtained as $\log q^*(x) = f(x) + C$. Since $q^* \in \mathcal{P}$, we have $q^*(x) = \exp(f(x) - A(f)) = p_f(x)$, which is the second conclusion. Plugging the optimal solution to $l(q)$, we obtain the maximum as $\log \int_\Omega \exp(\lambda f(x))\, dx$, which is exactly $A(f)$, leading to the first conclusion. ■

B Variants of Dynamics Embedding

Besides the Hamiltonian Embedding we introduced in the main text, we can also embed alternative dynamics.

B.1 Generalized Hamiltonian Embedding

As $T \to \infty$, neural network composed by the leapfrog operation can approximate arbitrary smooth distributions on continuous variables. However, with the finite step, the vanilla HMC may still not flexible enough. Levy et al. [2017] generalizes the HMC for better flexibility in finite step case by introducing more learnable components and combining the variable partition architecture [Dinh et al., 2016], which leads to a new embedded neural network layer, *i.e.*, $(x', v') = \mathbf{L}(x, v)$ as

$$\begin{aligned}
v^{\frac{1}{2}} &= v \odot \exp\left(\frac{\eta}{2} S_v(\xi_1(x))\right) - \frac{\eta}{2}\left(-\nabla_x f(x) \odot \exp(\eta Q_v(\xi_1(x))) + T(\xi_1(x))\right), \\
x^{\frac{1}{2}} &= x_{m_t} + m_t \odot \left[x \odot \exp(\eta S_x(\xi_2(m_t))) + \eta\left(v^{\frac{1}{2}} \odot \exp(\eta Q_x(\xi_2(m_t)))\right)\right], \\
x' &= x^{\frac{1}{2}}_{\bar{m}_t} + \bar{m}_t \odot \left[x^{\frac{1}{2}} \odot \exp(\eta S_x(\xi_2(\bar{m}_t))) + \eta\left(v^{\frac{1}{2}} \odot \exp(\eta Q_x(\xi_2(\bar{m}_t)))\right)\right], \\
v' &= v \odot \exp\left(\frac{\eta}{2} S_v(\xi_1(x'))\right) - \frac{\eta}{2}\left(-\nabla_x f(x) \odot \exp(\eta Q_v(\xi_1(x'))) + T(\xi_1(x'))\right),
\end{aligned}$$

where $\xi_1(x) = (x, -\nabla_x f(x), t)$ and $\xi_2(m) = (m \odot x, v, t)$. The $m \in \{0,1\}^d$ and $\bar{m} = 1 - m$. The (S_v, Q_v, T_v) and (S_x, Q_x, T_x) are the learnable functions. Although the new operation $\mathbf{L}$ is no longer shear mapping, the Jacobian matrix is diagonal whose determinant is easy to be computed. Specifically, we list the determinant of Jacobians of the transformation as below: **i)**, from (x, v) to $\left(x, v^{\frac{1}{2}}\right)$ is $\exp\left(\frac{\eta}{2}\mathbf{1} \cdot S_v(\xi_1(x))\right)$, from $\left(x, v^{\frac{1}{2}}\right)$ to $\left(x^{\frac{1}{2}}, v^{\frac{1}{2}}\right)$ is $\exp(\eta m_t \cdot S_x(\xi_2(m_t)))$, from $\left(x^{\frac{1}{2}}, v^{\frac{1}{2}}\right)$ to $\left(x', v^{\frac{1}{2}}\right)$ is $\exp(\eta \bar{m}_t \cdot S_x(\xi_2(\bar{m}_t)))$, from $\left(x', v^{\frac{1}{2}}\right)$ to (x', v') is $\exp\left(\frac{\eta}{2}\mathbf{1} \cdot S_v(\xi_1(x'))\right)$.

We can use the modified $\mathbf{L}$ to neural network and plug into (7) to learn the sampler as well as the exponential family.

B.2 Deterministic Langevin Embedding

The Hamiltonian dynamics requires an auxiliary variable v with the same size of the x, doubling the memory cost. We embed the *deterministic Langevin dynamics* to form $x' = \mathbf{L}_{f,M}(x)$ as $x' = x + \eta \nabla_x f(x)$ with $x^0 \sim q_\theta^0(x)$. By the change-of-variable rule, we have $q_{f,M}^T\left(x^T\right) = q_\theta^0(x_0) \prod_{t=1}^T \left|\det \frac{\partial x^t}{\partial x^{t-1}}\right|$. Plug such parametrization of the dual distribution into (5), we achieve the alternative objective

$$\max_{f \in \mathcal{F}} \min_{\theta, M, \eta} \ell(f; \theta, M, \eta) := \widehat{\mathbb{E}}_{\mathcal{D}}[f] - \mathbb{E}_{x^0 \sim q_\theta^0(x)}\left[f\left(x^T\right) - \log q_\theta^0(x) - \sum_{t=1}^T \log\left|\det \frac{\partial x^t}{\partial x^{t-1}}\right|\right]. \tag{9}$$

For the log-determinant term, $\log\left|\det \frac{\partial x^t}{\partial x^{t-1}}\right| = \log\left|\det\left(I + \eta \mathbf{H}^f(x^t)\right)\right|$, where $\mathbf{H}^f_{i,j} = \frac{\partial^2 f(x)}{\partial x_i \partial x_j}$. Then, the gradient $\frac{\partial \log\left|\det\left(I + \eta \mathbf{H}^f(x^t)\right)\right|}{\partial f} = \eta \operatorname{tr}\left(\left(I + \eta \mathbf{H}^f(x_t)\right)^{-1} \frac{\partial H^f(x^t)}{\partial f}\right)$. However, the computation of the log-determinant and its derivative w.r.t. f are expensive. We can apply the polynomial expansion to approximate it.

Denoting δ as the bound of the spectrum of $\mathbf{H}^f(x^t)$ and $C := \frac{\eta\delta}{1+\eta\delta} I - \frac{1}{1+\eta\delta}\mathbf{H}^f(x^t)$, we have $\lambda(C) \in (-1, 1)$. Then,

$$\log\left|\det\left(I + \eta\mathbf{H}^f\left(x^t\right)\right)\right| = d\log(1+\eta\delta) + \operatorname{tr}(\log(I - C)).$$

We can apply Taylor expansion or Chebyshev expansion to approximate the $\operatorname{tr}(\log(I-C))$. Specifically, we have

- Stochastic Taylor Expansion [Boutsidis et al., 2017] Recall $\log(1-x) = -\sum_{k=1}^{\infty}\frac{x^k}{k}$, we have the Taylor expansion

  $$\operatorname{tr}(\log(I - C)) = -\sum_{i=1}^{k}\frac{\operatorname{tr}\left(C^i\right)}{i}.$$

  To avoid the matri-matrix multiplication, we further approximate the $\operatorname{tr}(C) = \mathbb{E}_z\left[z^\top C z\right]$ with z as Rademacher random variables, *i.e.*, Bernoulli distribution with $p = \frac{1}{2}$.

  Particularly, if we set $i = 1$, recall the $\operatorname{tr}\left(\mathbf{H}^f(x)\right) = \nabla_x^2 f(x)$, we can directly calculate without the Hutchinson approximation.
- Stochastic Chebyshev Expansion [Han et al., 2015] We can approximate with Chebyshev polynomial, *i.e.*,

  $$\operatorname{tr}(\log(I - C)) = \sum_{i=1}^{k} c_i \operatorname{tr}(R_i(C)),$$

  where $R(\cdot)$ denotes the Chebshev polynomial as $R_i(x) = 2xR_{i-1}(x) - R_{i-2}(x)$ with $R_1(x) = x$ and $R_0(x) = 1$. The $c_i = \frac{2}{k+1}\sum_{j=0}^{k}\log(1-s_j)R_i(s_j)$ if $i \geqslant 1$, otherwise $c_0 = \frac{1}{n+1}\sum_{j=0}^{k}\log(1-s_j)$ where $s_j = \cos\left(\frac{\pi\left(k+\frac{1}{2}\right)}{k+1}\right)$ for $j = 0, 1, \ldots, k$.

  Similarly, we can use the Hutchinson approximation to avoid matrix-matrix multiplication.

B.3 Continuous-time Langevin Embedding

We discuss several discretized dynamics embedding above. In this section, we take the continuous-time limit $\eta \to 0$ in the deterministic Langevin dynamics, *i.e.*, $\frac{dx}{dt} = \nabla_x f(x)$. Follow the change-of-variable rule, we obtain

$$\begin{aligned} & q(x') = p(x)\det(I + \eta\nabla_x f(x)) \\ \Rightarrow \quad & \log q(x') - \log p(x) = -\operatorname{tr}\log\left(I + \eta\mathbf{H}^f(x)\right) = -\eta\nabla_x^2 f(x) + \mathcal{O}\left(\eta^2\right). \end{aligned}$$

As $\eta \to 0$, we have

$$\frac{d\log q(x,t)}{dt} = -\nabla_x^2 f(x). \tag{10}$$

Remark (connections to Fokker-Planck equation): Consider the $\frac{dx}{dt} = \nabla_x f(x)$ as a SDE with zero diffusion term, by Fokker-Planck equation, we obtain the PDE w.r.t. $q(x,t)$ as

$$\frac{\partial q(x,t)}{\partial t} = -\nabla\cdot\left(\nabla_x f(x)\, q(x,t)\right).$$

Alternatively, we can also derive the (10) from the Fokker-Planck equation by explicitly writing the derivative. Specifically,

$$\begin{aligned} \frac{dq(x,t)}{dt} &= \frac{\partial q(x,t)}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial q(x,t)}{\partial t} \\ &= \frac{\partial q(x,t)}{\partial x}\nabla_x f(x) - \nabla\cdot\left(\nabla_x f(x)\, q(x,t)\right) \\ &= \frac{\partial q(x,t)}{\partial x}\nabla_x f(x) - \nabla_x^2 f(x)\, q(x,t) - \nabla_x f(x)\frac{\partial q(x,t)}{\partial t} \\ &= -\nabla_x^2 f(x)\, q(x,t). \end{aligned}$$

Therefore, we have

$$\frac{1}{q(x,t)}\frac{dq(x,t)}{dt} = -\nabla_x^2 f(x) \Rightarrow \left[\begin{array}{c} \frac{d\log q(x,t)}{dt} = -\nabla_x^2 f(x) \\ \frac{dx}{dt} = \nabla_x f(x) \end{array}\right]. \tag{11}$$

Based on (10), we can obtain the samples and its density value by

$$\begin{bmatrix} x^t \\ \log q\left(x^t\right) - \log p_\theta^0\left(x^0\right) \end{bmatrix} = \int_{t_0}^{t_1} \begin{bmatrix} \nabla_x f\left(x\left(t\right)\right) \\ -\nabla_x^2 f\left(x(t)\right) \end{bmatrix} dt := \mathbf{L}_{f,t_0,t_1}\left(x\right). \tag{12}$$

We emphasize that this dynamics is different from the continuous-time flow proposed in Grathwohl et al. [2018], where we have $\nabla_x^2 f\left(x\right)$ in the ODE rather than a trace operator, which requires one more Hutchinson stochastic approximation. We noticed that Zhang et al. [2018] also exploits the Monge-Ampère equation to design the flow-based model for unsupervised learning. However, their learning algorithm is totally different from ours. They use the parameterization as a new flow and fit the model by matching a *separate* distribution; while in our case, the exponential family and flow share the same parameters and match each other automatically.

We can approximate the integral using a numerical quadrature methods. One can approximate the $\nabla_{(f,t_0,t_1)}\ell\left(f;t_0,t_1\right)$ by the derivative through the numerical quadrature. Alternatively, we denote $g\left(t\right) = -\frac{\partial \ell(f,t_0,t_1)}{\partial x(t)}$, by the adjoint method, the $\frac{\ell(f,t_0,t_1)}{\partial f}$ is also characterized by ODE

$$\frac{\partial \ell\left(f,t_0,t_1\right)}{\partial f} = \int_{t_0}^{t_1} -g\left(t\right)^\top \nabla_f \cdot \nabla_x f\left(x\right) dt, \tag{13}$$

and can be approximated by numerical quadrature too.

We can combine the discretized and continuous-time Langevin dynamics by simply stacking several layers of $\mathbf{L}_{f,t_0,t_1}$.