Conditional Noise-Contrastive Estimation of Energy-Based Models by Jumping Between Modes

Hanlin Yu
University of Helsinki

Michael U. Gutmann
University of Edinburgh

Arto Klami
University of Helsinki

Omar Chehab
Carnegie Mellon University

Abstract

Learning Energy-Based Models (EBMs) is notoriously difficult when the data distribution is multi-modal. Standard methods such as Score Matching — even when amortized across many noisy versions of the data as in Energy-Based Diffusion Models — often fail to capture relative energies between modes because they rely solely on local energy differences. We address this limitation by also considering global energy differences. To do so, we use Conditional Noise-Contrastive Estimation (CNCE) which estimates energy differences between pairs of points drawn using a freely chosen noise distribution. We design this noise distribution to propose pairs of points from different modes, thus comparing the modes directly. We further obtain the asymptotic estimation error of CNCE, derive a theoretically optimal noise distribution, and provide a practical algorithm that combines local and global energy differences. Experiments show that this approach substantially improves estimation in multi-modal settings.

1 Introduction

Energy-Based Models (EBMs) of data are a powerful way of using neural networks for density estimation [Murphy, 2023, Rhodes, 2023]. Such models parameterize the unnormalized density by an energy function that can be an expressive neural network. Such modelling flexilbity comes at a price: the normalizing factor integrates the neural network over the data space which is usually intractable to compute. This has led to the development of many methods for estimating the parameters that avoid or approximate the normalizing factor.

Many standard methods for estimating the parameters of an EBM are sample-inefficient when data is multi-modal [Koehler et al., 2023]. Intuitively, this is because they use *local* differences to learn the *global* energy function. This is for example the case for Score Matching losses and variants which estimate the *gradient* of the energy function. Local differences in the energy function can be estimated with high precision even if the energy function is globally distorted (change the relative weights of two far-away modes) [Wenliang and Kanagawa, 2021].

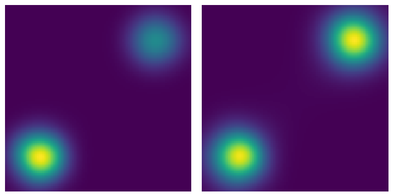
Figure 1: Learnt densities after training an EBM on a bi-modal Gaussian mixture with equal weights using CNCE. Left: solely using local noises may not learn the correct scalings. Right: combining local and global transitions resolves the multimodal issue.

To address the issue of learning EBMs of multi-modal data, we use Conditional Noise-Contrastive Estimation (CNCE) [Ceylan and Gutmann, 2018] to estimate *local and global* energy differences and provide a theoretical

EurIPS 2025 Workshop on Principles of Generative Modeling (PriGM).

result that motivates this choice. Our experiments show that this approach substantially improves EBM estimation in multi-modal settings.

2 Background

Formally, an Energy-Based Model is written as $p_{\boldsymbol{\theta}}(\boldsymbol{x}) = f_{\boldsymbol{\theta}}(\boldsymbol{x})/Z(\boldsymbol{\theta})$, where the unnormalized density $f_{\boldsymbol{\theta}} = \exp(-E_{\boldsymbol{\theta}}(\boldsymbol{x}))$ is parameterized by an energy function $E_{\boldsymbol{\theta}}$ that can be a neural network.

Score Matching losses Learning an EBM can be achieved by minimizing a Score Matching loss [Hyvärinen, 2005], motivated by the observation that Stein scores $\nabla \log p_{\boldsymbol{\theta}}(\boldsymbol{x})$ are convenient to learn as the intractable normalizing factor cancels out. A popular version is [Vincent, 2011]

$$\mathcal{L}_{\mathrm{DSM}}(\boldsymbol{\theta}) = \mathbb{E}_{p_d(\boldsymbol{x})p_n(\boldsymbol{y}|\boldsymbol{x},\epsilon)}\left[\|\nabla \log p_n(\boldsymbol{y}|\boldsymbol{x},\epsilon) - \nabla \log p_{\boldsymbol{\theta}}(\boldsymbol{y})\|^2\right], \tag{1}$$

which is termed the Denoising Score Matching loss. Here, $\boldsymbol{y}$ is a version of the data that is corrupted by Gaussian noise $p_n(\boldsymbol{y}|\boldsymbol{x}, t) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{x}, \epsilon \boldsymbol{I})$ for a given ϵ, and the loss is consistent as $\epsilon \to 0$. There is extensive empirical [Wenliang and Kanagawa, 2021, Zhang et al., 2022] and theoretical [Koehler et al., 2023] evidence that such losses are sample-inefficient for learning EBMs of multi-modal data, essentially because the score is a local quantity. The following attempts to bypass this problem.

Amortizing Score Matching losses across noise levels Recently, Score Matching losses have been amortized over different corruption levels of the data ϵ, as in Energy-Based Diffusion Models (EBM Diffusion) [Song et al., 2021, Du et al., 2023, Thornton et al., 2025]

$$\mathcal{L}_{\mathrm{EBM-Diffusion}}(\boldsymbol{\theta}) = \mathbb{E}_{p(\epsilon)p_d(\boldsymbol{x})p_n(\boldsymbol{y}|\boldsymbol{x},\epsilon)}\left[\|\nabla \log p_n(\boldsymbol{y}|\boldsymbol{x},\epsilon) - \nabla \log p_{\boldsymbol{\theta}}(\boldsymbol{y}|\epsilon)\|^2\right]. \tag{2}$$

Note that in Eq. 2, the parameterization is shared across noise levels ϵ. The hope is that higher noise levels ϵ where the score estimation is easier will benefit lower noise levels ϵ where the score estimation is harder. Indeed, recent theory [Qin and Risteski, 2024] suggests that minimizing Score Matching losses over different noise levels as in Eq. 2 *does* improve over a single noise level as in Eq. 1, and can even be Fisher-efficient [Chewi et al., 2025].

In practice, however, estimating parameters by minimizing this loss still incurs a high error, as we later verify in Figure 2. This error has a practical impact, given the usage of EBM Diffusion for compositional generation for example [Du et al., 2023]. This has led recent works to suggest regularizing the amortized loss Eq. 2 with an extra loss [Guth et al., 2025, Aggarwal et al., 2025, He et al., 2025, Plainer et al., 2025]. This extra loss term is often itself amortized over noise levels and these works do not provide a theory for why it improves the estimation.

Conditional Noise-Contrastive Estimation loss We next review Conditional Noise-Contrastive Estimation (CNCE) [Ceylan and Gutmann, 2018]. It learns the differences in energy $\log p_{\boldsymbol{\theta}}(\boldsymbol{y}) - \log p_{\boldsymbol{\theta}}(\boldsymbol{x})$ for any $\boldsymbol{x}$ and $\boldsymbol{y}$, which also cancels out the intractable normalizing constant.

$$\mathcal{L}_{\mathrm{CNCE}}(\boldsymbol{\theta}) = -2\mathbb{E}_{p_d(\boldsymbol{x})p_n(\boldsymbol{y}|\boldsymbol{x})}\left[\log \sigma\big(\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) - \log p_{\boldsymbol{\theta}}(\boldsymbol{y}) + \log p_n(\boldsymbol{y}|\boldsymbol{x}) - \log p_n(\boldsymbol{x}|\boldsymbol{y})\big)\right], \tag{3}$$

where σ is the sigmoid function. When $\boldsymbol{x}$ and $\boldsymbol{y}$ are neighboring points, this difference approximates the Stein Score. The CNCE loss also uses a conditional noise distribution p_n which determines the jump from clean data $\boldsymbol{x}$ to corrupted data $\boldsymbol{y}$ used to compute the energy differences. Unlike the Score Matching loss in Eq. 1, CNCE is consistent for any choice of p_n, a free design choice.

Many choices of p_n have been explored in the literature [Olmin et al., 2024] — many of them result in a pair of *nearby* points $\boldsymbol{x}$ and $\boldsymbol{y}$. This locality implies that the estimation is only accurate when evaluated on points that are close to each other. This is the case when the noise distribution is obtained by adding a small Gaussian perturbation $p_n(\boldsymbol{y}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{x}, \epsilon \boldsymbol{I})$, the CNCE loss recovers the population Score Matching loss as $\epsilon \to 0$ [Ceylan and Gutmann, 2018, Theorem 1]. Another popular choice to adaptively update p_n, so that it takes takes local MCMC steps toward the current version of the model [Ceylan, 2017, Section 4.3] [Welling et al., 2003] which recovers the well-known Contrastive Divergence algorithm [Hinton, 2002, Yair and Michaeli, 2021].

However, for highly multimodal distributions, there may exist large *gaps* between the modes so training on *nearby* points may result in inaccurate energies. CNCE local is indeed prone to this problem, which we verify in the experiments section. Welling et al. [2003] propose a solution for the related Contrastive Divergence algorithm: they choose p_n to jump between modes of the data, but rely on local Gaussian approximations or Metropolis accept-reject steps that are inefficient in high dimensions. Ceylan and Gutmann [2018] choose p_n as a Gaussian perturbation whose standard deviation is tuned, but large values can easily lead to jumps in low-density regions of the data.

3 Solving the multimodality problem by jumping between modes

In this section, we formalize a solution to the multimodality problem from a theoretical and a practical perspective. We start by deriving a formula for the asymptotic estimation error of CNCE.

Theorem 1 (Estimation error of CNCE) *The CNCE estimator $\hat{\boldsymbol{\theta}}$ that minimizes the empirical version of the loss in Eq. 3 using N_d data points is asymptotically normal $\hat{\boldsymbol{\theta}} \sim \mathcal{N}(\boldsymbol{\theta}^*, \boldsymbol{\Sigma})$ where $\boldsymbol{\Sigma} = \frac{1}{N_d}\boldsymbol{C}_1^{-1}\boldsymbol{C}_2\boldsymbol{C}_1^{-1}$ and its mean-squared error is*

$$\|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*\|^2 = \frac{1}{N_d}\mathrm{trace}(\boldsymbol{C}_1^{-1}\boldsymbol{C}_2\boldsymbol{C}_1^{-1}) + o\big(\frac{1}{N_d}\big). \tag{4}$$

The matrices are functions of the log density differences $F_\theta(\boldsymbol{x}, \boldsymbol{y}) = \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) - \log p_{\boldsymbol{\theta}}(\boldsymbol{y}) + \log p_n(\boldsymbol{y}|\boldsymbol{x}) - \log p_n(\boldsymbol{x}|\boldsymbol{y})$

$$\boldsymbol{C}_1 := \mathbb{E}_{p_d(\boldsymbol{x})p_n(\boldsymbol{y}|\boldsymbol{x})}\left[\nabla_{\boldsymbol{\theta}} F_{\boldsymbol{\theta}^*}(\boldsymbol{x},\boldsymbol{y})\nabla_{\boldsymbol{\theta}} F_{\boldsymbol{\theta}^*}(\boldsymbol{x},\boldsymbol{y})^\top \sigma(-F_{\boldsymbol{\theta}^*}(\boldsymbol{x},\boldsymbol{y}))\sigma(F_{\boldsymbol{\theta}^*}(\boldsymbol{x},\boldsymbol{y}))\right], \tag{5}$$

$$\boldsymbol{C}_2 := \mathbb{E}_{p_d(\boldsymbol{x})p_n(\boldsymbol{y}|\boldsymbol{x})}\left[\nabla_{\boldsymbol{\theta}} F_{\boldsymbol{\theta}^*}(\boldsymbol{x},\boldsymbol{y})\nabla_{\boldsymbol{\theta}} F_{\boldsymbol{\theta}^*}(\boldsymbol{x},\boldsymbol{y})^\top \sigma(-F_{\boldsymbol{\theta}^*}(\boldsymbol{x},\boldsymbol{y}))^2\right]. \tag{6}$$

We next show how to choose the *noise distribution* to reduce the above error. Indeed, if we had access to an oracle of the data density we wish to estimate, then our following result proves that setting the noise density equal to the data density is near-optimal. This is equivalent to randomly jumping between data points located among the various modes.

Corollary 2 (Near-optimal noise distribution) *The oracle noise $p_n(\boldsymbol{y}|\boldsymbol{x}) = p_d(\boldsymbol{x})$ is quasi Fisher-efficient, in that it achieves the Cramer-Rao Lower Bound, up to a constant factor of* 2*,*

$$\mathbb{E}\|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*\|^2 = \frac{2}{N_d}\mathrm{trace}(\boldsymbol{I}_F^{-1}) + o\big(\frac{1}{N_d}\big), \tag{7}$$

where $\boldsymbol{I}_F$ denotes the Fisher Information matrix.

In practice, we obviously do not have access to such an oracle of the data density as recommended by Corollary 2. But for many problems we can approximate it separately. While this introduces a new source of error, it is reasonable to hope it does not degrade the estimation guarantees [Uehara et al., 2020]. A popular choice is to take many MCMC steps toward the data distribution as in the popular Contrastive Divergence algorithm, which in theory remains quasi Fisher-efficient [Glaser et al., 2024] but in practice can be unstable to train [Carbone et al., 2023, Section 4.2]. Another choice is to use a non-parametric approximation the data distribution such as a Kernel Density Estimate (KDE) [Uehara et al., 2020].

Yet, formal guarantees on the quality of the empirical estimate are hard to obtain [Uehara et al., 2020]. In practice, we set the loss equal to a mixture of CNCE losses between (i) a loss with unconditional p_n given by a KDE approximation of the data distribution of two distributions so that CNCE estimates *global differences* in the energy and (ii) a loss with conditional p_n given by a small Gaussian perturbation so that CNCE estimates *local differences* in the energy. The mixture parameter interpolates between (i) the theoretically optimal choice given our Corollary 2 and (ii) what is used in practice, similar to Score Matching. We tune the mixture parameter as a hyperparameter.

4 Experiments

We consider training an EBM parameterized using a neural network to fit data generated from a 20 dimensional Gaussian mixture with 40 components. The means of the Gaussian mixture components are randomly sampled from the standard normal distribution, while the covariances are simply

Table 1: NCE algorithms for learning the 20 dimensional Gaussian mixture. CNCE mix is clearly the best among all considered variants.

Metric	NCE	NCE GMM	CNCE	CNCE mix	CNCE GMM
Ratio $\downarrow$	54.9 ± 8.6	39.1 ± 6.6	130.0 ± 43.4	23.8 ± 0.5	**20.2** $\pm$ 0.5
NLL $\downarrow$	3.5 ± 1.3	-5.9 ± 1.6	2.1 ± 2.7	**-6.6** $\pm$ 0.5	-3.3 ± 2.5
MSE $\downarrow$	334.2 ± 48.6	87.6 ± 29.7	329.2 ± 103.9	**66.2** $\pm$ 7.4	130.9 ± 55.8

given by $0.1^2\boldsymbol{I}$. We evaluate the quality of the resulting models by (i) the mean squared error of the estimated log density ratio between two data samples against the ground truth (*ratio*), (ii) the normalized negative log likelihood on data points (*NLL*), (iii) the mean squared error of the normalized log density against the ground truth (*MSE*). The models are normalized using importance sampling with 10^5 samples, with the proposal distribution given by the ground truth data distribution.

Loss functions based on binary classification We consider the CNCE loss function with the different design choices: using local Gaussian jumps (*CNCE local*), global jumps between data points (*CNCE global*), and our proposed mixture of both (*CNCE mix*). We compare this with another class of loss functions for learning EBMs, called Noise-Contrastive Estimation (NCE) [Gutmann and Hyvärinen, 2012], which is obtained as a binary classification problem between data points and noise points. For NCE, we consider the following design choices: using standard Gaussian noise (*NCE*) and using a KDE approximation of the data as noise (*NCE global*) which is Fisher-efficient in theory [Uehara et al., 2020]. As shown in Table 1, CNCE mix achieves the best performance, significantly outperforming the baselines.

Regularizing the loss function used in Energy-Based Diffusion Models We consider another loss for training EBMs, as used in EBM Diffusion and recalled in Eq. 2 (*Diffusion*). Our code is based off Du et al. [2023]. Apart from the 20 dimensional Gaussian mixture as considered before, we utilize a two dimensional Gaussian mixture with 25 modes with equal weights based on Schröder et al. [2023] for better visualizations, whose means are distributed on a grid and the covariances are small. We further consider adding a regularization term to the loss, given by CNCE with global jumps (*Diffusion regularized*). As shown in Figure 2, EBM Diffusion in its raw form struggles to learn the correct distribution, failing to capture the relative energy scales between the modes. Adding the regularizer significantly improves upon the vanilla version. Due to differences in *e.g.* training complexities, the results may not be directly comparable against Table 1.

Metric	EBM diff	EBM diff reg
Ratio $\downarrow$	1783.6 ± 535.0	**103.5** $\pm$ 27.0
NLL $\downarrow$	31.0 ± 7.6	**-8.5** $\pm$ 0.3
MSE $\downarrow$	2972.5 ± 886.7	**80.6** $\pm$ 17.0

(a) EBM Diffusions for 20 dimensional Gaussian mixture.

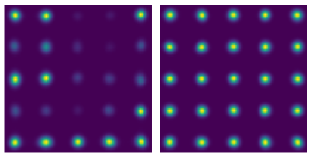
(b) EBM Diffusions for 2 dimensional Gaussian mixture. Left and right: without and with regularization.

Figure 2: The proposed regularization significantly reduces the estimation errors of EBM Diffusion.

5 Discussion

CNCE for learning Energy-Based Models As popularized by Diffusion Models [Song et al., 2021], Score Matching losses [Hyvärinen, 2005, Vincent, 2011] are enjoying widespread use in generative modeling. CNCE is closely related to Score Matching, as it recovers Score Matching in expectation for infinitesimal noise [Ceylan and Gutmann, 2018]. We argue that it is currently under-appreciated by the community. In our work, we analyze the estimation errors of CNCE and highlight the potential of utilizing it for a broader set of problems, such as regularizing EBM Diffusions.

Broader interest in CNCE Note that CNCE was originally formulated as minimizing a loss function that solves a certain binary classification task. Versions of the binary task above have been used in the machine learning literature, without necessarily parameterizing the ratio by an Energy-Based Model (EBM) in place of p_d. One example is to detect the reversibility of a timeseries [Agrawal et al., 2022]. Another example comes from computer vision, where the task is to predict the permutation of a (longer) input tuple that has been scrambled: this is likened to 'solving a puzzle' [Noroozi and Favaro, 2016]. This can be formalized as CNCE, where label $k = 1$ indexes the original tuple order and label $k = 2$ indexes the permutation. Hence, our results may be of independent interest given the renewed interest in versions of CNCE, while its estimation error was unavailable.

Acknowledgments and Disclosure of Funding

HY and AK acknowledge the research environment provided by ELLIS Institute Finland, and were supported by the Research Council of Finland Flagship programme: Finnish Center for Artificial Intelligence FCAI. MUG was supported in part by Gen AI - the AI Hub for Generative Models, funded by EPSRC (EP/Y028805/1). The authors wish to acknowledge CSC - IT Center for Science, Finland, for computational resources.

References

R. Aggarwal, J. Chen, N. M. Boffi, and D. Koes. BoltzNCE: Learning likelihoods for boltzmann generation with stochastic interpolants and noise contrastive estimation. In *ICML 2025 Generative AI and Biology (GenBio) Workshop*, 2025.

M. N. Agrawal, H. Lang, M. Offin, L. Gazit, and D. A. Sontag. Leveraging time irreversibility with order-contrastive pre-training. In *International Conference on Artificial Intelligence and Statistics, AISTATS 2022*, volume 151 of *Proceedings of Machine Learning Research*. PMLR, 2022.

D. Carbone, M. Hua, S. Coste, and E. Vanden-Eijnden. Efficient training of energy-based models using jarzynski equality. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

C. Ceylan. Conditional noise-contrastive estimation: With application to natural image statistics. Master's thesis, KTH, School of Computer Science and Communication (CSC), Stockholm, Sweden, 2017. Independent thesis, advanced level (20 credits / 30 HE credits).

C. Ceylan and M. U. Gutmann. Conditional Noise-Contrastive Estimation of Unnormalised Models. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 726–734. PMLR, July 2018.

S. Chewi, A. Kalavasis, A. Mehrotra, and O. Montasser. DDPM score matching is asymptotically efficient. In *ICLR 2025 Workshop on Deep Generative Model in Machine Learning: Theory, Principle and Efficacy*, 2025.

Y. Du, C. Durkan, R. Strudel, J. B. Tenenbaum, S. Dieleman, R. Fergus, J. Sohl-Dickstein, A. Doucet, and W. S. Grathwohl. Reduce, Reuse, Recycle: Compositional Generation with Energy-Based Diffusion Models and MCMC. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 8489–8510. PMLR, July 2023.

P. Glaser, K. H. Huang, and A. Gretton. Near-optimality of contrastive divergence algorithms. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 91036–91090. Curran Associates, Inc., 2024.

F. Guth, Z. Kadkhodaie, and E. P. Simoncelli. Learning normalized image densities via dual score matching, 2025. _eprint: 2506.05310.

M. U. Gutmann and A. Hyvärinen. Noise-Contrastive Estimation of Unnormalized Statistical Models, with Applications to Natural Image Statistics. *Journal of Machine Learning Research*, 13 (11):307–361, 2012.

J. He, J. M. Hernández-Lobato, Y. Du, and F. Vargas. RNE: plug-and-play diffusion inference-time control and energy-based training, 2025. _eprint: 2506.05668.

K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.

G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Comput.*, 14(8):1771–1800, Aug. 2002. ISSN 0899-7667.

A. Hyvärinen. Estimation of Non-Normalized Statistical Models by Score Matching. *Journal of Machine Learning Research*, 6(24):695–709, 2005.

D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

F. Koehler, A. Heckett, and A. Risteski. Statistical efficiency of score matching: The view from isoperimetry. In *The Eleventh International Conference on Learning Representations*, 2023.

K. P. Murphy. *Probabilistic Machine Learning: Advanced Topics*. MIT Press, 2023.

M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, 2016.

A. Olmin, J. Lindqvist, L. Svensson, and F. Lindsten. On the connection between noise-contrastive estimation and contrastive divergence. In S. Dasgupta, S. Mandt, and Y. Li, editors, *International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 3016–3024. PMLR, 02–04 May 2024.

M. Plainer, H. Wu, L. Klein, S. Günnemann, and F. Noé. Consistent Sampling and Simulation: Molecular Dynamics with Energy-Based Diffusion Models, 2025. URL `https://arxiv.org/abs/2506.17139`. _eprint: 2506.17139.

Y. Qin and A. Risteski. Fit like you sample: Sample-efficient generalized score matching from fast mixing diffusions. In S. Agrawal and A. Roth, editors, *Proceedings of Thirty Seventh Conference on Learning Theory*, volume 247 of *Proceedings of Machine Learning Research*, pages 4413–4457. PMLR, 30 Jun–03 Jul 2024.

B. Rhodes. *Advances in Scalable Learning and Sampling of Unnormalised Models*. PhD thesis, University of Edinburgh, Edinburgh, UK, 2023. PhD thesis.

T. Schröder, Z. Ou, J. Lim, Y. Li, S. Vollmer, and A. Duncan. Energy Discrepancies: A Score-Independent Loss for Energy-Based Models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 45300–45338. Curran Associates, Inc., 2023.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations*, 2021.

J. Thornton, L. Bethune, R. Zhang, A. Bradley, P. Nakkiran, and S. Zhai. Composition and control with distilled energy diffusion models and sequential monte carlo. In *AISTATS 2025*, 2025.

M. Uehara, T. Kanamori, T. Takenouchi, and T. Matsuda. A unified statistically efficient estimation framework for unnormalized models. In *International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 809–819. PMLR, 26–28 Aug 2020.

A. W. v. d. Vaart. *Asymptotic Statistics*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1998.

P. Vincent. A Connection Between Score Matching and Denoising Autoencoders. *Neural Computation*, 23(7):1661–1674, 2011.

M. Welling, A. Mnih, and G. E. Hinton. Wormholes Improve Contrastive Divergence. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16. MIT Press, 2003.

L. K. Wenliang and H. Kanagawa. Blindness of score-based methods to isolated components and mixing proportions, 2021. _eprint: 2008.10087.

O. Yair and T. Michaeli. Contrastive Divergence Learning is a Time Reversal Adversarial Game. In *International Conference on Learning Representations*, 2021.

M. Zhang, O. Key, P. Hayes, D. Barber, B. Paige, and F.-X. Briol. Towards Healing the Blindness of Score Matching. In *NeurIPS 2022 Workshop on Score-Based Methods*, 2022.

Appendix

The appendix is organized as follows.

A Conditional Noise-Contrastive Estimation (CNCE)

Data points sampled from a data distribution p_d can be used to define a binary classification task called Conditional Noise-Contrastive Estimation (CNCE) [Ceylan and Gutmann, 2018]. A tuple $(\boldsymbol{x}, \boldsymbol{y})$ is associated with a binary label $k \in \{0, 1\}$. The tuple is generated from either generative probabilities

$$p(\boldsymbol{x}, \boldsymbol{y}|k = 1) = p_d(\boldsymbol{x})p_n(\boldsymbol{y}|\boldsymbol{x}), \quad p(\boldsymbol{x}, \boldsymbol{y}|k = 2) = p_d(\boldsymbol{y})p_n(\boldsymbol{x}|\boldsymbol{y}) \tag{8}$$

with class priors $p(k = 1) = p(k = 2) = 0.5$, where p_n is a data-dependent noise distribution (e.g. Gaussian blur). The distribution of class 1 can be sampled by picking a data point x and then noising it into y. The distribution of class 2 can be sampled by permuting these inputs. Bayes' rule yields the discriminative probabilities

$$p(k = 1|\boldsymbol{x}, \boldsymbol{y}) = \frac{p_d(\boldsymbol{x})p_n(\boldsymbol{y}|\boldsymbol{x})}{p_d(\boldsymbol{x})p_n(\boldsymbol{y}|\boldsymbol{x}) + p_d(\boldsymbol{y})p_n(\boldsymbol{x}|\boldsymbol{y})} = \text{sigmoid}\Big(\log \frac{p_d(\boldsymbol{x})p_n(\boldsymbol{y}|\boldsymbol{x})}{p_d(\boldsymbol{y})p_n(\boldsymbol{x}|\boldsymbol{y})}\Big), \tag{9}$$

$$p(k = 2|\boldsymbol{x}, \boldsymbol{y}) = \frac{p_d(\boldsymbol{y})p_n(\boldsymbol{x}|\boldsymbol{y})}{p_d(\boldsymbol{x})p_n(\boldsymbol{y}|\boldsymbol{x}) + p_d(\boldsymbol{y})p_n(\boldsymbol{x}|\boldsymbol{y})} = \text{sigmoid}\Big(-\log \frac{p_d(\boldsymbol{x})p_n(\boldsymbol{y}|\boldsymbol{x})}{p_d(\boldsymbol{y})p_n(\boldsymbol{x}|\boldsymbol{y})}\Big), \tag{10}$$

which are explicit functions of the reversibility ratio $\frac{p_d(\boldsymbol{x})p_n(\boldsymbol{y}|\boldsymbol{x})}{p_d(\boldsymbol{y})p_n(\boldsymbol{x}|\boldsymbol{y})}$ that can be parameterized and learnt. In short, CNCE detects reversibility — a key concept in the dynamics of physical systems.

CNCE loss and hyperparameters The logistic classification loss for CNCE is

$$\mathcal{L}(\theta) = -\mathbb{E}_{\boldsymbol{x},\boldsymbol{y}\sim p_1} \log \sigma(F(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})) - \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}\sim p_0} \log \sigma(-F(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})) \tag{11}$$

where $F_\theta(\boldsymbol{x}, \boldsymbol{y}) = \log \frac{p(\boldsymbol{x};\boldsymbol{\theta})p_n(\boldsymbol{y}|\boldsymbol{x})}{p(\boldsymbol{y};\boldsymbol{\theta})p_n(\boldsymbol{x}|\boldsymbol{y})}$. Exploiting the (unusual) symmetry of the problem stated in Lemma 4 allows us to rewrite both sums as one

$$\mathcal{L}(\theta) = -\mathbb{E}_{\boldsymbol{x},\boldsymbol{y}\sim p_1} \log \sigma(F(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})) - \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}\sim p_1} \log \sigma(F(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})) = -2\mathbb{E}_{\boldsymbol{x},\boldsymbol{y}\sim p_1} \log \sigma(F(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})) \tag{12}$$

While this is all equal for an infinite number of samples, we anticipate discretizing this loss and joining both sums as one exploits the method of Common Random Numbers, known in statistics for variance reduction [1]. This yields

$$\mathcal{L}_T(\theta) = \frac{-2}{N_d K} \sum_{i=1}^{N_d} \sum_{j=1}^{K} \log \sigma(F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta})), \tag{13}$$

introducing by the choice of discretization, a different number of data and noise samples with $K = \frac{N_n}{N_d}$.

[1]This is, in fact, a key difference with the finite-sample NCE loss, which keeps two distinct sums.

B Theoretical results

B.1 Useful lemmas

The Fisher Information Matrix plays a special role: under certain assumptions, its inverse provides the best possible estimation error for $\boldsymbol{\theta}$. We therefore wish to recognize it when performing calculations for unnormalized models. In practice, we are interested in $p(\boldsymbol{x};\boldsymbol{\theta}) = \frac{f(\boldsymbol{x};\boldsymbol{\theta})}{Z}$.

Lemma 3 (Fisher Information Matrix in terms of the unnormalized model) *The Fisher Information Matrix of the normalized model p can be written in terms of the unnormalized model f, as*

$$\begin{aligned} \boldsymbol{I}_F(\boldsymbol{\theta}) &= \mathrm{Var}_{\boldsymbol{x}\sim p}[\nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x};\boldsymbol{\theta})] && (14)\\ &:= \mathbb{E}_p\left[\nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x};\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x};\boldsymbol{\theta})^\top\right] - \mathbb{E}_p\left[\nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x};\boldsymbol{\theta})\right] \mathbb{E}_p[\nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x};\boldsymbol{\theta})]^\top. && (15)\end{aligned}$$

Proof. By definition, the Fisher Information matrix is

$$\boldsymbol{I}_F(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x}\sim p}\left[\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x};\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x};\boldsymbol{\theta})^\top\right]. \quad (16)$$

We then relate the Fisher score vector of the normalized model p to the unnormalized model f, as

$$\begin{aligned} \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{x},\boldsymbol{\theta}) &= \nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x};\boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}} \log \int f(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x} = \nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x};\boldsymbol{\theta}) - \frac{\nabla_{\boldsymbol{\theta}} \int f(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x}}{\int f(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x}} && (17)\\ &= \nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x};\boldsymbol{\theta}) - \int \frac{\nabla_{\boldsymbol{\theta}} f(\boldsymbol{x};\boldsymbol{\theta})}{\int f(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x}} d\boldsymbol{x} && (18)\\ &= \nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x};\boldsymbol{\theta}) - \int \nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x};\boldsymbol{\theta}) \frac{f(\boldsymbol{x};\boldsymbol{\theta})}{\int f(\boldsymbol{x};\boldsymbol{\theta}) d\boldsymbol{x}} d\boldsymbol{x} && (19)\\ &= \nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x};\boldsymbol{\theta}) - \mathbb{E}_{\boldsymbol{x}\sim p} \nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x};\boldsymbol{\theta}) && (20)\end{aligned}$$

Plugging this back into Eq. 16 yields the result. □

Lemma 4 (Symmetry of CNCE loss) *Given one data / noise pair $(\boldsymbol{x},\boldsymbol{y})$, CNCE loss yields two terms that are exactly the same.*

Proof.

$$\log \sigma\left(\log \frac{f(\boldsymbol{x};\boldsymbol{\theta}) p_n(\boldsymbol{y}|\boldsymbol{x})}{f(\boldsymbol{y};\boldsymbol{\theta}) p_n(\boldsymbol{x}|\boldsymbol{y})}\right) = \log\left(1 - \sigma\left(\log \frac{f(\boldsymbol{y};\boldsymbol{\theta}) p_n(\boldsymbol{x}|\boldsymbol{y})}{f(\boldsymbol{x};\boldsymbol{\theta}) p_n(\boldsymbol{y}|\boldsymbol{x})}\right)\right). \quad (21)$$

□

B.2 Proof of Theorem 1

Recall that under reasonable assumptions, the estimation error of a parameter is asymptotically given by the following formula [Vaart, 1998]

$$\|\hat{\boldsymbol{\theta}}_N - \boldsymbol{\theta}^*\|^2 = \mathrm{trace}(\boldsymbol{\Sigma}) + o\left(\frac{1}{N}\right), \quad (22)$$

where $\boldsymbol{\Sigma}$ is known as the asymptotic covariance matrix defined as

$$\boldsymbol{\Sigma} = \mathbb{E}\left[\boldsymbol{H}_N(\boldsymbol{\theta}^*)\right]^{-1} \times \mathrm{Var}\left[J_N(\boldsymbol{\theta}^*)\right] \times \mathbb{E}\left[\boldsymbol{H}_N(\boldsymbol{\theta}^*)\right]^{-1}, \quad (23)$$

which is the product of three matrices involving the empirical loss function. In the following, we compute the relevant quantities that will define the asymptotic variance.

Empirical loss We recall the empirical loss function

$$\mathcal{L}_N(\boldsymbol{\theta}) = \frac{-2}{N_d K} \sum_{i=1}^{N_d} \sum_{j=1}^{K} \log \sigma(F(\boldsymbol{x}_i, \boldsymbol{y}_{ij};\boldsymbol{\theta})), \quad F(\boldsymbol{x},\boldsymbol{y};\boldsymbol{\theta}) = \log \frac{f(\boldsymbol{x};\theta) p_n(\boldsymbol{y}|\boldsymbol{x})}{f(\boldsymbol{y};\boldsymbol{\theta}) p_n(\boldsymbol{x}|\boldsymbol{y})}. \quad (24)$$

Gradient of the empirical loss We can compute the gradient of the empirical loss function

$$\boldsymbol{J}_N(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} \mathcal{L}_N(\boldsymbol{\theta}) = \frac{-2}{N_d K} \sum_{i=1}^{N_d} \sum_{j=1}^{K} \frac{1}{\sigma(F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}, \boldsymbol{\theta}))} \nabla_{\boldsymbol{\theta}} \sigma(F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta})) \tag{25}$$

$$= \frac{-2}{N_d K} \sum_{i=1}^{N_d} \sum_{j=1}^{K} \frac{1}{\sigma(F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}, \boldsymbol{\theta}))} \sigma^{'}(F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta})) \nabla_{\boldsymbol{\theta}} F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta}) \tag{26}$$

$$= \frac{-2}{N_d K} \sum_{i=1}^{N_d} \sum_{j=1}^{K} \nabla_{\boldsymbol{\theta}} F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}, \boldsymbol{\theta}) \ \ (1 - \sigma(F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta}))) \tag{27}$$

$$= \frac{-2}{N_d K} \sum_{i=1}^{N_d} \sum_{j=1}^{K} \nabla_{\boldsymbol{\theta}} F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}, \boldsymbol{\theta}) \ \ P(Y=0 | (\boldsymbol{x}_i, \boldsymbol{y}_{ij}); \boldsymbol{\theta}) \tag{28}$$

$$= \frac{-2}{N_d K} \sum_{i=1}^{N_d} \sum_{j=1}^{K} \boldsymbol{s}(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta}), \tag{29}$$

using that $\sigma^{'}(.) = \sigma(.)(1-\sigma(.))$. In the last line, we abbreviated the summand of the gradient with $\boldsymbol{s}$.

Hessian of the empirical loss We can also compute the Hessian of the empirical loss function

$$\boldsymbol{H}_N(\boldsymbol{\theta}) = \nabla^2_{\boldsymbol{\theta}} \mathcal{L}_N(\boldsymbol{\theta}) = \frac{-2}{N_d K} \sum_{i=1}^{N_d} \sum_{j=1}^{K} \nabla^2_{\boldsymbol{\theta}} F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta}) \ \ (1 - \sigma(F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta}))) \tag{30}$$

$$- \nabla_{\boldsymbol{\theta}} F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta})^\top \ \ \sigma(F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta}))\,(1 - \sigma(F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta}))) \tag{31}$$

$$= \frac{-2}{N_d K} \sum_{i=1}^{N_d} \sum_{j=1}^{K} \nabla^2_{\boldsymbol{\theta}} F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta}) \ \ P(Y=0 | (\boldsymbol{x}_i, \boldsymbol{y}_{ij}); \boldsymbol{\theta}) \tag{32}$$

$$- \nabla_{\boldsymbol{\theta}} F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} F(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta})^\top P(Y=0 | (\boldsymbol{x}_i, \boldsymbol{y}_{ij}); \boldsymbol{\theta}) P(Y=1 | (\boldsymbol{x}_i, \boldsymbol{y}_{ij}); \boldsymbol{\theta}) \tag{33}$$

$$= \frac{-2}{N_d K} \sum_{i=1}^{N_d} \sum_{j=1}^{K} \boldsymbol{S}(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta}), \tag{34}$$

where in the last line we abbreviate the summand of the Hessian with $\boldsymbol{S}$.

Expected Hessian of the empirical loss We can now compute the expected Hessian of the empirical loss.

$$\mathbb{E}\left[\boldsymbol{H}_N(\boldsymbol{\theta})\right] = \frac{-2}{N_d K} \mathbb{E}_{\boldsymbol{x}_i \sim p_d, \boldsymbol{y}_{ij} \sim p_n(.|\boldsymbol{x}_i)} \left[\sum_{i=1}^{N_d} \sum_{j=1}^{K} \boldsymbol{S}(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta}) \right] \tag{35}$$

$$= \frac{-2}{K} \mathbb{E}_{\boldsymbol{x} \sim p_d, \boldsymbol{y}_j \sim p_n(.|\boldsymbol{x})} \left[\sum_{j=1}^{K} \boldsymbol{S}(\boldsymbol{x}, \boldsymbol{y}_j; \boldsymbol{\theta}) \right] = -2 \mathbb{E}_{\boldsymbol{x} \sim p_d, \boldsymbol{y} \sim p_n(.|\boldsymbol{x})} \left[\boldsymbol{S}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) \right] \ . \tag{36}$$

When unpacking the summand $\boldsymbol{S}$, the first term is null as a result of the symmetry of the problem by Lemma 4. This leaves

$$\mathbb{E}\left[\boldsymbol{H}_N(\theta)\right] = 2\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y}) \sim p_1} \left[\nabla_{\boldsymbol{\theta}} F(\boldsymbol{x}, \boldsymbol{y}; \theta) \nabla_{\boldsymbol{\theta}} F(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})^\top \ \ P(Y=0|(\boldsymbol{x},\boldsymbol{y}); \boldsymbol{\theta}) P(Y=1|(\boldsymbol{x},\boldsymbol{y}); \boldsymbol{\theta}) \right] \tag{37}$$

$$= 2\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y}) \sim p_1} \Big[\left(\nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x}; \boldsymbol{\theta}^*) - \nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{y}; \boldsymbol{\theta}^*) \right) \left(\nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{x}; \boldsymbol{\theta}^*) - \nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{y}; \boldsymbol{\theta}^*) \right)^\top \tag{38}$$

$$P(Y=0|(\boldsymbol{x},\boldsymbol{y}); \boldsymbol{\theta}) P(Y=1|(\boldsymbol{x},\boldsymbol{y}); \boldsymbol{\theta}) \Big], \tag{39}$$

where we unpacked the log density ratio $F(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\theta})$.

Variance of the gradient of the empirical loss We can also compute the variance of the gradient of the empirical loss

$$\mathrm{Var}\boldsymbol{J}_N(\theta) = \frac{4}{N_d^2 K^2}\mathrm{Var}_{\boldsymbol{x}_i\sim p_d, \boldsymbol{y}_{ij}\sim p_n(.|\boldsymbol{x}_i)}\left[\sum_{i=1}^{N_d}\sum_{j=1}^{K}\boldsymbol{s}(\boldsymbol{x}_i, \boldsymbol{y}_{ij}; \boldsymbol{\theta})\right] \tag{40}$$

$$= \frac{4}{N_d K^2}\mathrm{Var}_{\boldsymbol{x}\sim p_d, \boldsymbol{y}_j\sim p_n(.|\boldsymbol{x})}\left[\sum_{j=1}^{K}\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}_j; \boldsymbol{\theta})\right] \tag{41}$$

$$= \frac{4}{N_d K^2}\mathbb{E}_{\boldsymbol{x}\sim p_d, \boldsymbol{y}_j\sim p_n(.|\boldsymbol{x})}\left[\sum_{j=1}^{K}\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}_j; \boldsymbol{\theta})\sum_{j=1}^{K}\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}_j; \boldsymbol{\theta})^\top\right] \tag{42}$$

$$- \frac{4}{N_d K^2}\mathbb{E}_{\boldsymbol{x}\sim p_d, \boldsymbol{y}_j\sim p_n(.|\boldsymbol{x})}\left[\sum_{j=1}^{K}\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}_j; \boldsymbol{\theta})\right]\mathbb{E}_{\boldsymbol{x}\sim p_d, \boldsymbol{y}_j\sim p_n(.|\boldsymbol{x})}\left[\sum_{j=1}^{K}\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}_j; \boldsymbol{\theta})\right]^\top. \tag{43}$$

The two last terms are actually null given the empirical gradient is unbiased. Hence the variance of the gradient is

$$\mathrm{Var}\boldsymbol{J}_N(\theta) = \frac{4}{N_d K^2}\mathbb{E}_{\boldsymbol{x}\sim p_d, \boldsymbol{y}_j\sim p_n(.|\boldsymbol{x})}\left[\sum_{j=1}^{K}\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}_j; \boldsymbol{\theta})\sum_{j=1}^{K}\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}_j; \boldsymbol{\theta})^\top\right] \tag{44}$$

$$= \frac{4}{N_d K^2}\mathbb{E}_{\boldsymbol{x}\sim p_d, \boldsymbol{y}_j\sim p_n(.|\boldsymbol{x})}\left[\sum_{j=1}^{K}\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}_j; \boldsymbol{\theta})\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}_j; \boldsymbol{\theta})^\top\right] \tag{45}$$

$$+ \frac{4}{N_d K^2}\mathbb{E}_{\boldsymbol{x}\sim p_d, \boldsymbol{y}_j\sim p_n(.|\boldsymbol{x})}\left[\sum_{j\neq k}\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}_j; \boldsymbol{\theta})\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}_k; \boldsymbol{\theta})^\top\right] \tag{46}$$

$$= \frac{4K}{N_d K^2}\mathbb{E}_{\boldsymbol{x}\sim p_d, \boldsymbol{y}\sim p_n(.|\boldsymbol{x})}\left[\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})^\top\right] \tag{47}$$

$$\tag{48}$$

$$+ \frac{4(K^2-K)}{N_d K^2}\mathbb{E}_{\boldsymbol{x}\sim p_d, \boldsymbol{y}\sim p_n(.|\boldsymbol{x}), \boldsymbol{y}'\sim p_n(.|\boldsymbol{x})}\left[\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}'; \boldsymbol{\theta})^\top\right] \tag{49}$$

$$\tag{50}$$

$$= \frac{4K}{N_d K^2}\mathbb{E}_{\boldsymbol{x}\sim p_d, \boldsymbol{y}\sim p_n(.|\boldsymbol{x})}\left[\nabla_{\boldsymbol{\theta}}F(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})\nabla_{\boldsymbol{\theta}}F(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})^\top \;\; P(Y=0|(\boldsymbol{x}, \boldsymbol{y}); \boldsymbol{\theta})^2\right] \tag{51}$$

$$\tag{52}$$

$$+ \;\; \frac{4(K^2-K)}{N_d K^2}\mathbb{E}_{\boldsymbol{x}\sim p_d}\Big[\mathbb{E}_{\boldsymbol{y}\sim p_n(.|\boldsymbol{x})}\left[\nabla_\theta F(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})P(Y=0|(\boldsymbol{x}, \boldsymbol{y}); \boldsymbol{\theta})\right] \tag{53}$$

$$E_{\boldsymbol{y}'\sim p_n(.|\boldsymbol{x})}\left[\nabla_\theta F(\boldsymbol{x}, \boldsymbol{y}'; \boldsymbol{\theta})P(Y=0|(\boldsymbol{x}, \boldsymbol{y}'); \boldsymbol{\theta})\right]^\top\Big], \tag{54}$$

where in the last line we unpacked the summand $\boldsymbol{s}$ of the gradient.

Special case when $K = 1$ In particular, when $K = 1$ we have

$$\mathbb{E}\left[\boldsymbol{H}_N(\theta)\right] = 2\mathbb{E}_{(\boldsymbol{x}, \boldsymbol{y})\sim p_1}\left[\nabla_{\boldsymbol{\theta}}F(\boldsymbol{x}, \boldsymbol{y}; \theta)\nabla_{\boldsymbol{\theta}}F(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})^\top \;\; P(Y=0|(\boldsymbol{x}, \boldsymbol{y}); \boldsymbol{\theta})P(Y=1|(\boldsymbol{x}, \boldsymbol{y}); \boldsymbol{\theta})\right], \tag{55}$$

and

$$\mathrm{Var}\boldsymbol{J}_N(\theta) = \frac{4}{N_d}\mathbb{E}_{\boldsymbol{x}\sim p_d, \boldsymbol{y}_j\sim p_n(.|\boldsymbol{x})}\left[\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})^\top\right] \tag{56}$$

$$= \frac{4}{N_d}\mathbb{E}_{\boldsymbol{x}\sim p_d, \boldsymbol{y}_j\sim p_n(.|\boldsymbol{x})}\left[\nabla_{\boldsymbol{\theta}}F(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\theta})\nabla_{\boldsymbol{\theta}}F(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\theta})^\top P(Y=0|(\boldsymbol{x}, \boldsymbol{y}); \boldsymbol{\theta})^2\right]. \tag{57}$$

Denote

$$\boldsymbol{C}_1 = \mathbb{E}_{\boldsymbol{x}\sim p_d, \boldsymbol{y}_j \sim p_n(.|\boldsymbol{x})} \left[\nabla_{\boldsymbol{\theta}} F(\boldsymbol{x},\boldsymbol{y},\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} F(\boldsymbol{x},\boldsymbol{y},\boldsymbol{\theta})^\top P(Y=0|(\boldsymbol{x},\boldsymbol{y});\boldsymbol{\theta}) P(Y=1|(\boldsymbol{x},\boldsymbol{y});\boldsymbol{\theta})\right], \tag{58}$$

$$\boldsymbol{C}_2 = \mathbb{E}_{\boldsymbol{x}\sim p_d, \boldsymbol{y}_j \sim p_n(.|\boldsymbol{x})} \left[\nabla_{\boldsymbol{\theta}} F(\boldsymbol{x},\boldsymbol{y},\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} F(\boldsymbol{x},\boldsymbol{y},\boldsymbol{\theta})^\top P(Y=0|(\boldsymbol{x},\boldsymbol{y});\boldsymbol{\theta})^2\right]. \tag{59}$$

We have

$$\mathbb{E}\left[\boldsymbol{H}_N(\theta)\right] = 2\boldsymbol{C}_1, \tag{60}$$

$$\mathrm{Var}\boldsymbol{J}_N(\theta) = \frac{4}{N_d}\boldsymbol{C}_2, \tag{61}$$

$$\begin{aligned}\boldsymbol{\Sigma} &= \mathbb{E}\left[\boldsymbol{H}_N(\boldsymbol{\theta}^*)\right]^{-1} \times \mathrm{Var}\big[J_N(\boldsymbol{\theta}^*)\big] \times \mathbb{E}\left[\boldsymbol{H}_N(\boldsymbol{\theta}^*)\right]^{-1} &(62)\\ &= \frac{1}{2}\boldsymbol{C}_1^{-1}\frac{4}{N_d}\boldsymbol{C}_2\frac{1}{2}\boldsymbol{C}_1 &(63)\\ &= \frac{1}{N_d}\boldsymbol{C}_1^{-1}\boldsymbol{C}_2\boldsymbol{C}_1^{-1}. &(64)\end{aligned}$$

B.3 Proof of Corollary 2

With oracle noise, $p_n(\boldsymbol{y}|\boldsymbol{x}) = p_d(\boldsymbol{y})$ and $p_n(\boldsymbol{x}|\boldsymbol{y}) = p_d(\boldsymbol{x})$, and we have

$$P(Y=0|(\boldsymbol{x},\boldsymbol{y});\boldsymbol{\theta}^*) = P(Y=1|(\boldsymbol{x},\boldsymbol{y});\boldsymbol{\theta}^*) = 1/2, \tag{65}$$

therefore the expected Hessian becomes

$$\mathbb{E}\left[\boldsymbol{H}_N(\boldsymbol{\theta})\right] = \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_d,\boldsymbol{y}\sim p_d}\left[(\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{x};\boldsymbol{\theta}^*) - \nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{y};\boldsymbol{\theta}^*))(\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{x};\boldsymbol{\theta}^*) - \nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{y};\boldsymbol{\theta}^*))^\top\right] \quad , \tag{66}$$

and the variance of the gradient becomes

$$\begin{aligned}\mathrm{Var}\boldsymbol{J}_N(\boldsymbol{\theta}) &= \frac{1}{N_d}\mathbb{E}_{\boldsymbol{x}\sim p_d,\boldsymbol{y}\sim p_d}\left[\nabla_{\boldsymbol{\theta}}F(\boldsymbol{x},\boldsymbol{y};\boldsymbol{\theta})\nabla_{\boldsymbol{\theta}}F(\boldsymbol{x},\boldsymbol{y};\boldsymbol{\theta})^\top\right] &(67)\\ &= \frac{1}{N_d}\mathbb{E}_{\boldsymbol{x}\sim p_d,\boldsymbol{y}\sim p_d}\left[(\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{x};\boldsymbol{\theta}^*) - \nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{y};\boldsymbol{\theta}^*))(\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{x};\boldsymbol{\theta}^*) - \nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{y};\boldsymbol{\theta}^*))^\top\right] \quad . &(68)\end{aligned}$$

Therefore, the asymptotic variance matrix is given by

$$\boldsymbol{\Sigma} = \frac{4}{N_d}\mathbb{E}_{\boldsymbol{x}\sim p_d,\boldsymbol{y}\sim p_d}\left[(\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{x};\boldsymbol{\theta}^*) - \nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{y};\boldsymbol{\theta}^*))(\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{x};\boldsymbol{\theta}^*) - \nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{y};\boldsymbol{\theta}^*))^\top\right]^{-1} \quad . \tag{69}$$

Observe that

$$\begin{aligned}&\mathbb{E}_{\boldsymbol{x}\sim p_d,\boldsymbol{y}\sim p_d}\left[(\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{x};\boldsymbol{\theta}^*) - \nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{y};\boldsymbol{\theta}^*))(\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{x};\boldsymbol{\theta}^*) - \nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{y};\boldsymbol{\theta}^*))^\top\right] &(70)\\ &= \mathbb{E}_{\boldsymbol{x}\sim p_d,\boldsymbol{y}\sim p_d}\left[\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{x};\boldsymbol{\theta}^*)\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{x};\boldsymbol{\theta}^*)^\top\right] - \mathbb{E}_{\boldsymbol{x}\sim p_d,\boldsymbol{y}\sim p_d}\left[\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{x};\boldsymbol{\theta}^*)\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{y};\boldsymbol{\theta}^*)^\top\right] &(71)\\ &- \mathbb{E}_{\boldsymbol{x}\sim p_d,\boldsymbol{y}\sim p_d}\left[\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{y};\boldsymbol{\theta}^*)\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{x};\boldsymbol{\theta}^*)^\top\right] + \mathbb{E}_{\boldsymbol{x}\sim p_d,\boldsymbol{y}\sim p_d}\left[\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{y};\boldsymbol{\theta}^*)\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{y};\boldsymbol{\theta}^*)^\top\right] &(72)\\ &= 2\mathbb{E}_{\boldsymbol{x}\sim p_d}\left[\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{x};\boldsymbol{\theta}^*)\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{x};\boldsymbol{\theta}^*)^\top\right] - 2\mathbb{E}_{\boldsymbol{x}\sim p_d,\boldsymbol{y}\sim p_d}\left[\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{x};\boldsymbol{\theta}^*)\nabla_{\boldsymbol{\theta}}\log f(\boldsymbol{y};\boldsymbol{\theta}^*)^\top\right] &(73)\\ &= 2\boldsymbol{I}_F(\boldsymbol{\theta}) \quad , &(74)\end{aligned}$$

using Lemma 3 which relates the Fisher Information Matrix to the unnormalized density $f(\boldsymbol{x};\boldsymbol{\theta})$.

As such, we have

$$\boldsymbol{\Sigma} = \frac{2}{N_d} \boldsymbol{I}_F(\boldsymbol{\theta})^{-1}, \tag{75}$$

and

$$\mathbb{E}\|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*\|^2 = \frac{2}{N_d} \text{trace}(\boldsymbol{I}_F^{-1}) + o\big(\frac{1}{N_d}\big), \tag{76}$$

as claimed.

C Experimental details

The training sets, the validation sets and the test sets all have 10000 data points. We use a batch size of 256 and train for 50000 iterations. The models are evaluated after every 2000 steps, and the results with the lowest losses (for NCE algorithms) / negative ELBO (for EBM Diffusions) on the validation sets are reported.

For all experiments, we use Adam optimizer [Kingma and Ba, 2015] and tune the learning rates between $1e-4$, $3e-4$, $1e-3$ and $3e-3$. We run each configuration for a single run using random seed 1, and run the best configurations for two extra runs using random seeds 2 and 3 and aggregate the results to obtain uncertainty estimates.

The employed neural network is a small ResNet [He et al., 2016] based on the code provided by Du et al. [2023] [2], where we largely reuse the structure. For NCE family of algorithms which do not model the intermediate distributions, we do not use time embeddings and change the output dimensionality of the final layer to 1.

C.1 Loss functions based on binary classification

For the Gaussian noise as used by NCE, we consider as means the empirical means of the training set or directly using zero means. In terms of the covariances, we consider using the empirical covariances of the training set, using identity matrix and using an isotropic covariance $\sigma^2 \boldsymbol{I}$ where σ is $\frac{1}{2}$ of the maximum distance between any two training data points.

The standard deviations σ of the Gaussian KDE in NCE global and CNCE global and the standard deviations of the Gaussian transition kernels for CNCE local are all tuned between 0.01, 0.1 and 1.0.

For CNCE mix, we tune both the standard deviations of the transition kernels and the Gaussian KDEs using the aforementioned grid, and additionally tune the proportion of noises based on local transitions among all noises between 0.9, 0.75 and 0.5.

C.2 Regularizing the loss function used in Energy-Based Diffusion Models

We interpret the model's outputs given $t = 0$ as the learned clean data distribution. Similar to the settings of CNCE mix, we form the regularization term based on a KDE approximation with σ tuned between 0.0, 0.01, 0.1 and 1.0. We additionally tune the regularization weight λ between 0.1, 1.0 and 10.0, and use as training loss $\mathcal{L}_{\text{SM}} + \lambda \mathcal{L}_{\text{CNCE}}$, where $\mathcal{L}_{\text{SM}}$ is the original Score Matching loss as employed by EBM Diffusion and $\mathcal{L}_{\text{CNCE}}$ is the CNCE loss with global jumps.

The two dimensional Gaussian mixture based on Schröder et al. [2023] uses a grid of 25 Gaussian distributions, with the x, y coordinates of the means between $[-2, -1, 0, 1, 2]$, and the covariance $0.1^2 \boldsymbol{I}$.

[2]ResNetDiffusionModel in `https://github.com/yilundu/reduce_reuse_recycle/blob/main/notebooks/simple_distributions.ipynb`.