Entropy-MCMC: Sampling from Flat Basins with Ease

Bolian Li LI4468@PURDUE.EDU
Ruqi Zhang RUQIZ@PURDUE.EDU
Department of Computer Science, Purdue University, USA

Editors: Sophia Sanborn, Christian Shewmake, Simone Azeglio, Nina Miolane

Abstract

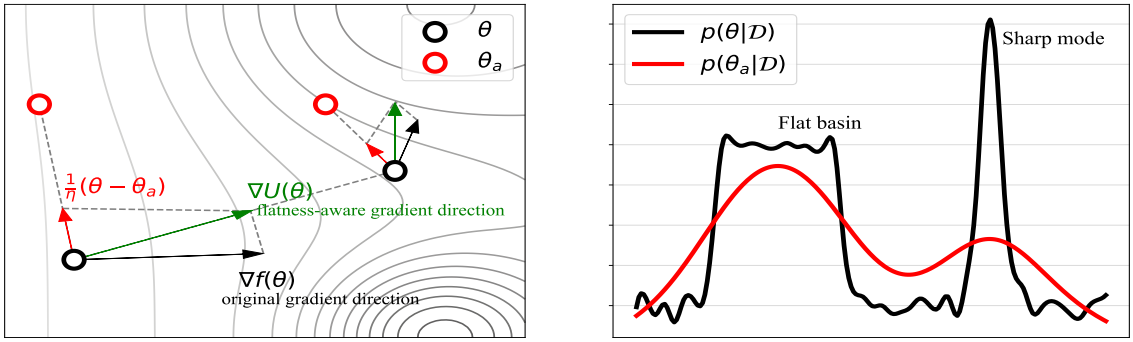
Bayesian deep learning counts on the quality of posterior distribution estimation. However, the posterior of deep neural networks is highly multi-modal in nature, with local modes exhibiting varying generalization performances. Given a practical budget, sampling from the original posterior can lead to suboptimal performances, as some samples may become trapped in "bad" modes and suffer from overfitting. Leveraging the observation that "good" modes with low generalization error often reside in flat basins of the energy landscape, we propose to bias the sampling on the posterior toward these flat regions. Specifically, we introduce an auxiliary guiding variable, the stationary distribution of which resembles a smoothed posterior free from sharp modes, to lead the MCMC sampler to flat basins. We prove the convergence of our method and further show that it converges faster than several existing flatness-aware methods in the strongly convex setting. Empirical results demonstrate that our method can successfully sample from flat basins of the posterior, and outperforms all compared baselines on multiple benchmarks including classification, calibration and out-of-distribution detection.

Keywords: Flatness-aware Learning, Bayesian Deep Learning, MCMC

1. Introduction

The effectiveness of Bayesian neural networks relies heavily on the quality of posterior distribution estimation. However, achieving an accurate estimation of the full posterior is extremely difficult due to its high-dimensional and highly multi-modal nature (Zhang et al., 2020b; Izmailov et al., 2021). Moreover, the numerous modes in the energy landscape typically exhibit varying generalization performances. Flat modes are often associated with enhanced accuracy and robustness, whereas sharp modes tend to have high generalization errors (Hochreiter and Schmidhuber, 1997; Keskar et al., 2017; Bahri et al., 2022). This connection between the geometry of energy landscape and generalization has spurred many works in optimization, ranging from theoretical understanding (Dziugaite and Roy, 2018; Jiang et al., 2019a) to new optimization algorithms (Izmailov et al., 2018; Foret et al., 2020).

However, most of the existing Bayesian methods are not aware of the flatness in the energy landscape during posterior inference (Welling and Teh, 2011). Their sampling strategies are usually low-energy-oriented and could not distinguish between flat and sharp modes that have the same energy values. This limitation can significantly undermine their generalization performance, particularly in the practical situations where capturing the full posterior is challenging. In light of this, we contend that we should prioritize capturing the flat modes when conducting posterior inference for Bayesian neural networks. This is advantageous for improved generalization and can also be rationalized from a Bayesian marginalization perspective. Within the flat basin, each model configuration occupies a substantial volume and



(*a*) Training dynamics at each step (*b*) Original and smoothed posteriors

Figure 1: (a) shows how the guiding variable $\boldsymbol{\theta}_a$ pulls $\boldsymbol{\theta}$ toward the flat basin; (b) shows two posterior distributions, where $p(\boldsymbol{\theta}_a|\mathcal{D})$ is a smoothed distribution transformed from $p(\boldsymbol{\theta}|\mathcal{D})$, and only keeps flat modes. Entropy-MCMC prioritizes flat modes by leveraging $\boldsymbol{\theta}_a$ from the smoothed posterior as a form of regularization.

contributes significantly to a more precise estimation of the predictive distribution. Moreover, existing flatness-aware methods often rely on a single solution to represent the entire flat basin (Foret et al., 2020), ignoring the fact that the flat basin contains many high-performing models. Therefore, Bayesian marginalization can potentially yield significant improvements by sampling from the flat basins (Wilson, 2020).

Prioritizing flat basins during posterior inference poses an additional challenge to Bayesian inference. Even for a single point, explicitly biasing toward the flat basins will introduce substantial computational overhead, inducing nested loops (Chaudhari et al., 2019; Dziugaite and Roy, 2018), doubled gradients calculation (Foret et al., 2020; Möllenhoff and Khan, 2022) or min-max problems (Foret et al., 2020). The efficiency problem needs to be addressed before any flatness-aware Bayesian method becomes practical.

In this paper, we propose an efficient MC sampling algorithm to explicitly prioritize flat basins in the energy landscape of deep neural networks. Specifically, we introduce an auxiliary guiding variable $\boldsymbol{\theta}_a$ into the Markov chain to pull the model $\boldsymbol{\theta}$ toward flat basins at each updating step (Fig. 1(*a*)). $\boldsymbol{\theta}_a$ is sampled from a smoothed posterior distribution which eliminates sharp modes based on local entropy (Baldassi et al., 2016) (Fig. 1(*b*)). Our method enjoys a simple joint distribution of $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_a$, and the computational overhead is similar to Stochastic gradient Langevin dynamics (SGLD) (Welling and Teh, 2011). Theoretically, we prove that our method is guaranteed to converge faster than some common flatness-aware methods (Chaudhari et al., 2019) in the strongly convex setting. Empirically, we demonstrate that our method successfully finds flat basins efficiently across multiple tasks.

Our main contributions are summarized as follows: i) we propose Entropy-MCMC (EMCMC), an MCMC method designed for sampling from flat basins in the energy landscape of deep neural networks. EMCMC utilizes an auxiliary guiding variable to efficiently steer the model toward flat basins; ii) we prove the convergence of EMCMC and further show that it converges faster than several existing flatness-aware methods in the strongly convex setting; iii) we provide extensive experimental results to demonstrate the advantages of EMCMC in sampling from flat basins. EMCMC outperforms all compared baselines on classification, calibration and out-of-distribution detection with comparable overhead akin to SGLD.

2. Methodology

We present the Entropy-MCMC algorithm. We introduce the guiding variable $\boldsymbol{\theta}_a$ obtained from the local entropy in section 2.1 and discuss the sampling strategy in section 2.2.

2.1. From Local Entropy to Flat Posterior

While flat basins in the energy landscape have been shown to be of good generalization (Bahri et al., 2022), finding such regions is still a problem due to the high-dimensional and multi-modal nature of DNN energy landscape. The updating direction of the model needs extra force to mitigate the risk of converging to sharp modes (Chaudhari et al., 2019; Foret et al., 2020). To bias sampling to flat basins, we look into the local entropy (Eq. 3), which can eliminate the sharp modes in the energy landscape (Chaudhari et al., 2019).

We begin by the original posterior $p(\boldsymbol{\theta}|\mathcal{D}) \propto \exp(-f(\boldsymbol{\theta})) = \exp\{\log p(\mathcal{D}|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta})\}$, which contains both sharp and flat modes. By replacing the original loss with local entropy, we obtain a smoothed posterior distribution in terms of a new variable $\boldsymbol{\theta}_a$: $p(\boldsymbol{\theta}_a|\mathcal{D}) \propto \exp \mathcal{F}(\boldsymbol{\theta}_a; \eta) = \int_{\boldsymbol{\Theta}} \exp\left\{-f(\boldsymbol{\theta}) - \frac{1}{2\eta}\|\boldsymbol{\theta} - \boldsymbol{\theta}_a\|^2\right\} d\boldsymbol{\theta}$. The effect of local entropy on this new posterior is visualized in Fig. 1(*b*). The new posterior measures both the depth and flatness of the mode in $p(\boldsymbol{\theta}|\mathcal{D})$ by considering surrounding energy values. Thereby, $p(\boldsymbol{\theta}_a|\mathcal{D})$ is expected to primarily capture flat modes in the energy landscape, which can be used as the desired external force to revise the updating directions of the model parameter $\boldsymbol{\theta}$.

However, the complex integral in $p(\boldsymbol{\theta}_a|\mathcal{D})$ requires marginalization on the model parameter $\boldsymbol{\theta}$, which poses a non-trivial challenge. Previous works using local entropy usually adopt an inner Markov chain with Monte Carlo (MC) approximation (Chaudhari et al., 2019; Dziugaite and Roy, 2018), which sacrifices the accuracy of local entropy computation and induces computationally expensive nested loops in training. We tackle this challenge in a simple yet principled manner, eliminating the need for nested loops or approximation. This is achieved by coupling $\boldsymbol{\theta} \sim p(\boldsymbol{\theta}|\mathcal{D})$ and $\boldsymbol{\theta}_a \sim p(\boldsymbol{\theta}_a|\mathcal{D})$ into a joint posterior distribution, which enjoys a simple form, as discussed in Lemma 1.

Lemma 1 *Assume* $\widetilde{\boldsymbol{\theta}} = [\boldsymbol{\theta}^T, \boldsymbol{\theta}_a^T]^T \in \mathbb{R}^{2d}$ *and* $\widetilde{\boldsymbol{\theta}}$ *has the following distribution:*

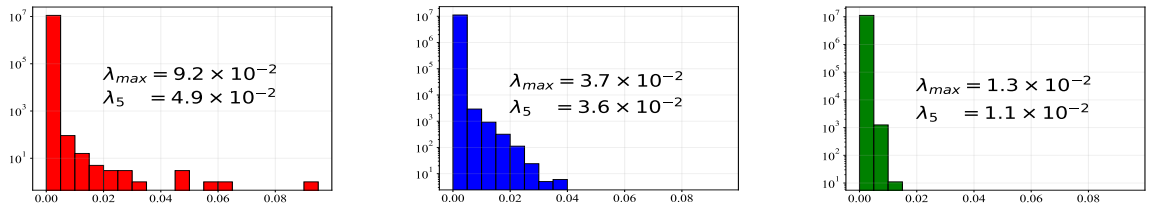
$$p(\widetilde{\boldsymbol{\theta}}|\mathcal{D}) = p(\boldsymbol{\theta}, \boldsymbol{\theta}_a|\mathcal{D}) \propto \exp\left\{-f(\boldsymbol{\theta}) - \frac{1}{2\eta}\|\boldsymbol{\theta} - \boldsymbol{\theta}_a\|^2\right\}. \tag{1}$$

Then the marginal distributions of $\boldsymbol{\theta}$ *and* $\boldsymbol{\theta}_a$ *are the original posterior* $p(\boldsymbol{\theta}|\mathcal{D})$ *and* $p(\boldsymbol{\theta}_a|\mathcal{D})$.

This joint posterior offers three key advantages: i) by coupling $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_a$, we avoid the intricate integral computation, and thus remove the requirement of expensive nested training loops and mitigate the MC approximation error; ii) the joint posterior turns out to be surprisingly simple, making it easy to obtain samples both empirically and theoretically (details discussed in Sections 2.2 and C); iii) after coupling, $\boldsymbol{\theta}_a$ provides additional paths for $\boldsymbol{\theta}$ to traverse, making $\boldsymbol{\theta}$ reach flat modes efficiently.

2.2. Sampling from Flat Basins

We discuss how to sample from the joint posterior distribution (Eq. 1) in this section. We adopt SGLD (Welling and Teh, 2011), a simple stochastic gradient MCMC algorithm that



(*a*) SGD (*b*) SGLD (*c*) EMCMC

Figure 2: Eigenspectrum of Hessian matrices of ResNet18 on CIFAR100. x-axis: eigenvalue, y-axis: frequency. A nearly all-zero eigenspectrum indicates a flat local mode. EMCMC successfully finds flat modes with smaller eigenvalues.

is suitable for deep neural networks, as the backbone of EMCMC sampling. More advanced MCMC algorithms can also be combined with our method. The energy function of the joint parameter variable $\widetilde{\boldsymbol{\theta}}$ is $U(\widetilde{\boldsymbol{\theta}}) = f(\boldsymbol{\theta}) + \frac{1}{2\eta}\|\boldsymbol{\theta} - \boldsymbol{\theta}_a\|^2$, and thus its gradients is given by:

$$\nabla_{\widetilde{\boldsymbol{\theta}}} U(\widetilde{\boldsymbol{\theta}}) = \begin{bmatrix} \nabla_{\boldsymbol{\theta}} U(\widetilde{\boldsymbol{\theta}}) \\ \nabla_{\boldsymbol{\theta}_a} U(\widetilde{\boldsymbol{\theta}}) \end{bmatrix} = \begin{bmatrix} \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) + \frac{1}{\eta}(\boldsymbol{\theta} - \boldsymbol{\theta}_a) \\ \frac{1}{\eta}(\boldsymbol{\theta}_a - \boldsymbol{\theta}) \end{bmatrix}. \tag{2}$$

The original gradient direction $\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$ is revised by $\frac{1}{\eta}(\boldsymbol{\theta} - \boldsymbol{\theta}_a)$ to get the flatness-aware gradient direction $\nabla_{\boldsymbol{\theta}} U(\widetilde{\boldsymbol{\theta}})$, shown in Fig. 1(*a*). Importantly, the practical implementation does not require computing $\nabla_{\boldsymbol{\theta}_a} U(\widetilde{\boldsymbol{\theta}})$ through back-propagation, as we can utilize the analytical expression presented in Eq. 2. Therefore, despite the $2d$ dimensions, our cost of gradient computation is essentially the *same* as d-dimensional models (e.g., standard SGLD).

With the form of the gradients of energy function, the training procedure of EMCMC is straightforward. The details are summarized in Algorithm 1. At testing stage, the collected samples $\mathcal{S}$ are used to approximate the predictive distribution $p(y|\boldsymbol{x}, \mathcal{D}) \approx \sum_{\boldsymbol{\theta}_s \in \mathcal{S}} p(y|\boldsymbol{x}, \boldsymbol{\theta}_s)$. Our choice of sampling from the joint posterior distribution using SGLD, rather than a Gibbs-like approach (Gelfand, 2000), is motivated by SGLD's ability to simultaneously update both $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_a$, which is more efficient (see Appendix E for a detailed discussion). For the sample set $\mathcal{S}$, we collect both $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_a$ after the burn-in period in order to obtain more high-quality and diverse samples in a given time budget.

In summary, thanks to EMCMC's simple joint distribution, conducting sampling in EMCMC is straightforward, and its computational cost is comparable to that of standard SGLD. Despite its simplicity, EMCMC is guaranteed to bias sampling to flat basins and obtain samples with enhanced generalization performances.

3. Experiments

The Hessian matrix measures the second-order gradients of a local mode on the energy landscape. Smaller eigenvalues of Hessian indicate a flatter local geometry (Foret et al., 2020). Since computing the exact Hessian of deep neural networks is extremely costly, we use the diagonal Fisher information matrix (Wasserman, 2004) to approximate its eigenspectrum: $[\lambda_1, ..., \lambda_d]^T \approx diag(\mathcal{I}(\boldsymbol{\theta})) = \mathbb{E}\left[(\nabla U - \mathbb{E}\nabla U)^2\right]$, where $\lambda_1, ..., \lambda_d$ are eigenvalues of the Hessian. Fig. 2 shows the eigenspectra of local modes discovered by different algorithms. The eigenvalues of EMCMC are much smaller compared with SGD and SGLD, indicating that the local geometry of EMCMC samples is flatter. The eigenspectrum comparison verifies the effectiveness of EMCMC to find and sample from flat basins.

References

Ahmad Ajalloeian and Sebastian U. Stich. On the convergence of sgd with biased gradients. *Journal of Machine Learning Research*, 2020. URL https://api.semanticscholar.org/CorpusID:234358812.

Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. An introduction to mcmc for machine learning. *Machine learning*, 50:5–43, 2003.

Dara Bahri, Hossein Mobahi, and Yi Tay. Sharpness-aware minimization improves language model generalization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7360–7371, 2022.

Carlo Baldassi, Christian Borgs, Jennifer T Chayes, Alessandro Ingrosso, Carlo Lucibello, Luca Saglietti, and Riccardo Zecchina. Unreasonable effectiveness of learning neural networks: From accessible states and robust ensembles to basic algorithmic schemes. *Proceedings of the National Academy of Sciences*, 113(48):E7655–E7662, 2016.

Devansh Bisla, Jing Wang, and Anna Choromanska. Low-pass filtering sgd for recovering flat optima in the deep learning optimization landscape. In *International Conference on Artificial Intelligence and Statistics*, pages 8299–8339. PMLR, 2022.

Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124018, 2019.

Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *International conference on machine learning*, pages 1683–1691. PMLR, 2014.

Adam D Cobb and Brian Jalaian. Scaling hamiltonian monte carlo inference for bayesian neural networks with symmetric splitting. In *Uncertainty in Artificial Intelligence*, pages 675–685. PMLR, 2021.

Arnak S Dalalyan and Avetik Karagulyan. User-friendly guarantees for the langevin monte carlo with inaccurate gradient. *Stochastic Processes and their Applications*, 129(12): 5278–5311, 2019.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *International Conference on Machine Learning*, pages 1019–1028. PMLR, 2017.

Gintare Karolina Dziugaite and Daniel Roy. Entropy-sgd optimizes the prior of a pac-bayes bound: Generalization properties of entropy-sgd and data-dependent priors. In *International Conference on Machine Learning*, pages 1377–1386. PMLR, 2018.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2020.

Alan E Gelfand. Gibbs sampling. *Journal of the American statistical Association*, 95(452): 1300–1304, 2000.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2018.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019a.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019b.

Sepp Hochreiter and Jürgen Schmidhuber. Simplifying neural nets by discovering flat minima. *Advances in neural information processing systems*, 7, 1994.

Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural computation*, 9(1):1–42, 1997.

P Izmailov, AG Wilson, D Podoprikhin, D Vetrov, and T Garipov. Averaging weights leads to wider optima and better generalization. In *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*, pages 876–885, 2018.

Pavel Izmailov, Sharad Vikram, Matthew D Hoffman, and Andrew Gordon Gordon Wilson. What are bayesian neural network posteriors really like? In *International conference on machine learning*, pages 4629–4640. PMLR, 2021.

Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2019a.

Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2019b.

Nitish Shirish Keskar, Jorge Nocedal, Ping Tak Peter Tang, Dheevatsa Mudigere, and Mikhail Smelyanskiy. On large-batch training for deep learning: Generalization gap and sharp minima. In *5th International Conference on Learning Representations, ICLR 2017*, 2017.

A Krizhevsky. Learning multiple layers of features from tiny images. *Master's thesis, University of Tront*, 2009.

Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Mu Li, Tong Zhang, Yuqiang Chen, and Alexander J Smola. Efficient mini-batch training for stochastic optimization. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 661–670, 2014.

Yi-An Ma, Tianqi Chen, and Emily Fox. A complete recipe for stochastic gradient mcmc. *Advances in neural information processing systems*, 28, 2015.

Dougal Maclaurin and Ryan P Adams. Firefly monte carlo: exact mcmc with subsets of data. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 4289–4295, 2015.

Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. *Advances in neural information processing systems*, 31, 2018.

Donna Katzman McClish. Analyzing a portion of the roc curve. *Medical decision making*, 9(3):190–195, 1989.

Thomas Möllenhoff and Mohammad Emtiyaz Khan. Sam as an optimal relaxation of bayes. In *The Eleventh International Conference on Learning Representations*, 2022.

Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29, 2015.

Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Saurabh Singh and Shankar Krishnan. Filter response normalization layer: Eliminating batch dependence in the training of deep neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11237–11246, 2020.

Larry Wasserman. *All of statistics: a concise course in statistical inference*, volume 26. Springer, 2004.

Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688, 2011.

Andrew Gordon Wilson. The case for bayesian deep learning. *arXiv preprint arXiv:2001.10995*, 2020.

Ruqi Zhang, A Feder Cooper, and Christopher M De Sa. Asymptotically optimal exact minibatch metropolis-hastings. *Advances in Neural Information Processing Systems*, 33: 19500–19510, 2020a.

Ruqi Zhang, Chunyuan Li, Jianyi Zhang, Changyou Chen, and Andrew Gordon Wilson. Cyclical stochastic gradient mcmc for bayesian deep learning. In *International Conference on Learning Representations*, 2020b.

Appendix A. Related Works

Flatness-aware Optimization. The concept of flatness in the energy landscape was first studied by Hochreiter and Schmidhuber (1994), and its connection with generalization was then empirically discussed by Keskar et al. (2017); Dinh et al. (2017); Jiang et al. (2019b). In order to pursue flatness for better generalization, Entropy-SGD (Chaudhari et al., 2019) introduced local entropy to consider the averaged loss of a region, SAM (Foret et al., 2020) developed a new optimizer to minimize the worst-case near the current model, bSAM (Möllenhoff and Khan, 2022) further improved SAM with a Bayes optimal convex lower bound, LPF (Bisla et al., 2022) introduced low-pass filter to actively search flat basins, and SWA (Izmailov et al., 2018) found that averaging weights along the trajectory of SGD training can also find flatter modes. Our Entropy-MCMC follows the local entropy measurement and collects more than a single point to fully exploit the flat basins.

MCMC on Deep Neural Networks. Markov chain Monte Carlo is a class of general and practical sampling algorithms (Andrieu et al., 2003), which has been applied to infer Bayesian neural network posteriors (Neal, 2012). SGMCMC (Welling and Teh, 2011; Ma et al., 2015) methods use the mini-batching technique to adapt MCMC to deep neural networks. SGHMC (Chen et al., 2014) exploited the second-order Langevin dynamics to calibrate the stochastic estimates of HMC gradients. cSGMCMC (Zhang et al., 2020b) further improves sampling efficiency by leveraging a cyclical stepsize schedule. Symmetric Split HMC (Cobb and Jalaian, 2021) developed a way to apply HMC to deep neural networks without stochastic gradients. Our Entropy-MCMC builds upon the SGMCMC framework and is designed to favor the flat basins in the energy landscape during sampling.

Appendix B. Preliminaries

Flatness-aware Optimization. One common flatness-aware optimization technique is to use the concept of local entropy, which measures the geometric properties of the energy landscape (Baldassi et al., 2016; Chaudhari et al., 2019). The local entropy is computed by:

$$\mathcal{F}(\boldsymbol{\theta};\eta) = \log \int_{\boldsymbol{\Theta}} \exp\left\{-f(\boldsymbol{\theta}') - \frac{1}{2\eta}\|\boldsymbol{\theta}-\boldsymbol{\theta}'\|^2\right\} d\boldsymbol{\theta}', \tag{3}$$

where $f(\cdot)$ is the loss function and η is a scalar. The local entropy of a point $\boldsymbol{\theta}$ is determined by its neighbors weighted by their distances, which considers the volume of local modes. Previous optimization methods minimize $-\mathcal{F}(\boldsymbol{\theta};\eta)$ to find the flat minimum.

SGMCMC. Given a dataset $\mathcal{D}$, neural networks with parameters $\boldsymbol{\theta} \in \mathbb{R}^d$ and a prior distribution $p(\boldsymbol{\theta})$, we can use Markov chain Monte Carlo (MCMC) to sample from the posterior $p(\boldsymbol{\theta}|\mathcal{D}) \propto \exp(-U(\boldsymbol{\theta}))$, where the energy function is $U(\boldsymbol{\theta}) = -\sum_{x\in\mathcal{D}} \log p(x|\boldsymbol{\theta}) - \log p(\boldsymbol{\theta})$. However, the computational cost for MCMC with large-scale data is too high to be practical. SGMCMC tackles this problem by stochastic gradient $\nabla U_{\boldsymbol{\Xi}}$ based on a subset of data $\boldsymbol{\Xi} \subseteq \mathcal{D}$. We use Stochastic Gradient Langevin Dynamics (SGLD) (Welling and Teh, 2011) in the paper as the backbone MCMC algorithm, which has the following updating rule:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha\nabla_{\boldsymbol{\theta}} U_{\boldsymbol{\Xi}}(\boldsymbol{\theta}) + \sqrt{2\alpha}\cdot\boldsymbol{\epsilon}, \tag{4}$$

where α is the step size and $\boldsymbol{\epsilon}$ is standard Gaussian noise. Our method can also be implemented by other SGMCMC methods. During testing, Bayesian marginalization is performed to make predictions based on the sample set collected during MC sampling $\mathcal{S} = \{\boldsymbol{\theta}_j\}_{j=1}^{M}$ and the predictive distribution is obtained by $p(y|\boldsymbol{x}, \mathcal{D}) = \int p(y|\boldsymbol{x}, \boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{D}) d\boldsymbol{\theta} \approx \sum_{\boldsymbol{\theta} \in \mathcal{S}} p(y|\boldsymbol{x}, \boldsymbol{\theta})$.

Appendix C. Theoretical Analysis

In this section, we provide a theoretical analysis on the convergence rate of Entropy-MCMC and compare it with previous local-entropy-based methods including Entropy-SGD (Chaudhari et al., 2019) and Entropy-SGLD (Dziugaite and Roy, 2018) (used as a theoretical tool in the literature rather than a practical algorithm). We leverage the 2-Wasserstein distance bounds of SGLD, which assumes the target distribution to be smooth and strongly log-concave (Dalalyan and Karagulyan, 2019). While the target distribution in this case is unimodal, it still reveals the superior convergence rate of EMCMC compared with existing flatness-aware methods. We leave the theoretical analysis on non-log-concave distributions for future work. Specifically, we have the following assumptions for the loss function $f(\cdot)$ and stochastic gradients:

Assumption 1 *The loss function $f(\boldsymbol{\theta})$ in the original posterior distribution $\pi = p(\boldsymbol{\theta}|\mathcal{D}) \propto \exp(-f(\boldsymbol{\theta}))$ is M-smooth and m-strongly convex (i.e., $m\boldsymbol{I} \preceq \nabla^2 f(\boldsymbol{\theta}') \preceq M\boldsymbol{I}$).*

Assumption 2 *The variance of stochastic gradients is bounded by $\mathbb{E}[\|\nabla f(\boldsymbol{\theta}) - \nabla \tilde{f}(\boldsymbol{\theta})\|^2] \leq \sigma^2$ for some constant $\sigma > 0$.*

To establish the convergence analysis for EMCMC, we first observe that the smoothness and convexity properties of the joint posterior distribution $\pi_{\text{joint}}(\boldsymbol{\theta}, \boldsymbol{\theta}_a) = p(\boldsymbol{\theta}, \boldsymbol{\theta}_a|\mathcal{D})$ in Eq. 1 is the same as $p(\boldsymbol{\theta}|\mathcal{D})$, which is formally stated in Lemma 2.

Lemma 2 *If Assumption 1 holds and $m \leq 1/\eta \leq M$, then the energy function in the joint posterior distribution $\pi_{joint}(\boldsymbol{\theta}, \boldsymbol{\theta}_a) = p(\boldsymbol{\theta}, \boldsymbol{\theta}_a|\mathcal{D})$ is also M-smooth and m-strongly convex.*

With the convergence bound of SGLD established by Dalalyan and Karagulyan (2019), we derive the convergence bound for EMCMC in Theorem 3.

Theorem 3 *Under Assumptions 1 and 2, let μ_0 be the initial distribution and μ_K be the distribution obtained by EMCMC after K iterations. If $m \leq 1/\eta \leq M$ and the step size $\alpha \leq 2/(m+M)$, the 2-Wasserstein distance between μ_K and π_{joint} will have the following upper bound:*

$$\mathcal{W}_2(\mu_K, \pi_{joint}) \leq (1 - \alpha m)^K \cdot \mathcal{W}_2(\mu_0, \pi) + 1.65(M/m)(2\alpha d)^{1/2} + \frac{\sigma^2 (2\alpha d)^{1/2}}{1.65M + \sigma\sqrt{m}}. \quad (5)$$

Comparing Theorem 3 with the convergence bound of SGLD obtained by Dalalyan and Karagulyan (2019), the only difference is that the dimension d is doubled to $2d$. Theorem 3 implies that the convergence rate of EMCMC will have at most a minor slowdown by a constant factor compared to SGLD while ensuring sampling from flat basins.

In contrast, previous local-entropy-based methods often substantially slow down the convergence in order to bias toward flat basins. For example, consider Entropy-SGD (Chaudhari et al., 2019) which minimizes a flattened loss function:

$$f_{\text{flat}}(\boldsymbol{\theta}) = -\mathcal{F}(\boldsymbol{\theta};\eta) = -\log \int_{\boldsymbol{\Theta}} \exp\left\{-f(\boldsymbol{\theta}') - \frac{1}{2\eta}\|\boldsymbol{\theta}-\boldsymbol{\theta}'\|^2\right\} d\boldsymbol{\theta}'. \tag{6}$$

We discuss the convergence bound of Entropy-SGD in Theorem 4, which shows how the presence of the integral (and the nested Markov chain induced by it) slows down the convergence.

Theorem 4 *Consider running Entropy-SGD to minimize the flattened loss function $f_{flat}(\boldsymbol{\theta})$ under Assumptions 1 and 2. Assume the inner Markov chain runs L iterations and the 2-Wasserstein distance between the initial and target distributions is always bounded by κ. Let f^*_{flat} represent the global minimum value of $f_{flat}(\boldsymbol{\theta})$ and $E_t := \mathbb{E}f_{flat}(\boldsymbol{\theta}_t) - f^*_{flat}$. If the step size $\alpha \le 2/(m+M)$, then we have the following upper bound:*

$$E_K \le \left(1 - \frac{\alpha m}{1+\eta M}\right)^K \cdot E_0 + \frac{A(1+\eta M)}{2m}, \tag{7}$$

where $A^2 = (1-\alpha m)^L \cdot \kappa + 1.65\left(\frac{M+1/\eta}{m+1/\eta}\right)(\alpha d)^{1/2} + \frac{\sigma^2(\alpha d)^{1/2}}{1.65(M+1/\eta)+\sigma\sqrt{m+1/\eta}}$.

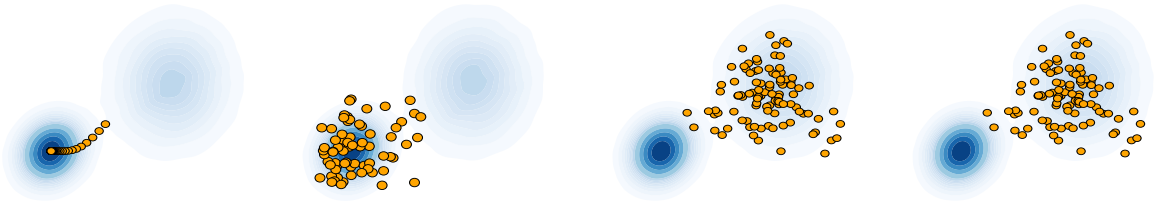
Another example is Entropy-SGLD (Dziugaite and Roy, 2018), a theoretical tool established to analyze Entropy-SGD. Its main distinction with Entropy-SGD is the SGLD updating instead of SGD updating in the outer loop. The convergence bound for Entropy-SGLD is established in Theorem 5.

Theorem 5 *Consider running Entropy-SGLD to sample from π_{flat} under Assumptions 1 and 2. Assume the inner Markov chain runs L iterations and the 2-Wasserstein distance between initial and target distributions is always bounded by κ. Let ν_0 be the initial distribution and ν_K be the distribution obtained by Entropy-SGLD after K iterations. If the step size $\alpha \le 2/(m+M)$, then*

$$\mathcal{W}_2(\nu_K, \pi_{flat}) \le (1-\alpha m)^K \cdot \mathcal{W}_2(\nu_0, \pi_{flat}) + 1.65\left(\frac{1+\eta M}{1+\eta m}\right)(M/m)(\alpha d)^{1/2} + \frac{A(1+\eta M)}{m}, \tag{8}$$

where $A^2 = (1-\alpha m)^L \cdot \kappa + 1.65\left(\frac{M+1/\eta}{m+1/\eta}\right)(\alpha d)^{1/2} + \frac{\sigma^2(\alpha d)^{1/2}}{1.65(M+1/\eta)+\sigma\sqrt{m+1/\eta}}$.

Comparing Theorem 3, 4 and 5, we observe that the convergence rates of Entropy-SGD and Entropy-SGLD algorithms are significantly hindered due to the presence of the nested Markov chains, which induces a large and complicated error term A. Since σ and α are typically very small, the third term in Theorem 3 will be much smaller than both the third term in Theorem 5 and the second term in Theorem 4. To summarize, the convergence rate of Entropy-MCMC is proved to be notably better than that of Entropy-SGD and Entropy-SGLD in the strongly convex setting.



(a) SGD (b) SGLD (c) EMCMC ($\boldsymbol{\theta}$) (d) EMCMC ($\boldsymbol{\theta}_a$)

Figure 3: Sampling trajectories on a synthetic energy landscape with sharp (lower left) and flat (top right) modes. The initial point is located at the ridge of two modes with no preference for either. EMCMC successfully bias toward the flat mode whereas SG(L)D is trapped in the sharp mode.

Appendix D. Experiments

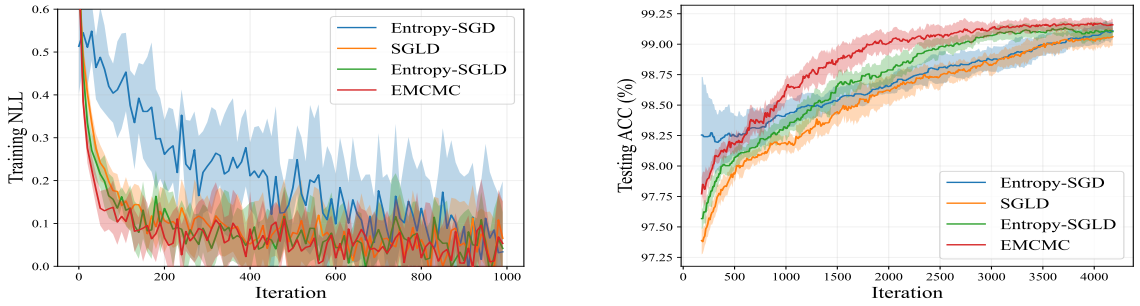
We conduct comprehensive experiments to show the superiority of EMCMC. Section D.1 and D.3 demonstrate that EMCMC can successfully sample from the flat basins. Section D.2 verifies the fast convergence of EMCMC. Section D.4 and D.5 demonstrate the outstanding performances of EMCMC on multiple benchmarks. Following Zhang et al. (2020b), we adopt a cyclical stepsize schedule for all sampling methods. For more implementation details, please refer to Appendix G.

D.1. Synthetic Examples

To test EMCMC's capability to sample from flat basins, we construct a two-mode energy landscape $\frac{1}{2}\mathcal{N}([-2,-1]^T, 0.5\boldsymbol{I}) + \frac{1}{2}\mathcal{N}([2,1]^T, \boldsymbol{I})$ to represent a sharp and a flat mode respectively. To make the case challenging, we set the initial point at $(-0.2, -0.2)$, the ridge of the two modes, which has no strong preference for either mode. Fig. 3 shows that the proposed EMCMC finds the flat basin while SGD and SGLD still prefer the sharp mode due to the slightly larger gradients coming from the sharp mode. From Fig. 3(c)&(d), we see that the samples of $\boldsymbol{\theta}_a$ are always around the flat mode, showing its ability to eliminate the sharp mode. Although $\boldsymbol{\theta}$ visits the sharp mode in the first few iterations, it subsequently inclines toward the flat mode, illustrating the influence of gradient revision by the guiding variable. It is noteworthy that the result of EMCMC is essentially *independent* of initialization with appropriate η, since the guiding variable $\boldsymbol{\theta}_a$ will always steer $\boldsymbol{\theta}$ to the flat mode. We show the results for different initialization in Fig. 6&7.

D.2. Logistic Regression

To verify the theoretical results on convergence rates in Section C, we conduct logistic regression on MNIST (LeCun, 1998) to compare EMCMC with Entropy-SGD Chaudhari et al. (2019), SGLD (Welling and Teh, 2011) and Entropy-SGLD (Dziugaite and Roy, 2018). We follow Maclaurin and Adams (2015) and Zhang et al. (2020a) to use a subset containing 7s and 9s and the resulting posterior is strongly log-concave, satisfying the assumptions in Section C. Fig. 4 shows that EMCMC converges faster than Entropy-SG(L)D, demonstrating



(a) (b)

Figure 4: Logistic regression on MNIST in terms of training NLL and testing accuracy (repeated 10 times). EMCMC converges faster than others, which is consistent with our theoretical analysis.

the advantage of using a simple joint distribution without the need for nested loops or MC approximation, which verifies Theorems 3& 4& 5. Besides, while EMCMC and SGLD share similar convergence rates, EMCMC achieves better generalization as shown by its higher test accuracy. This suggests that EMCMC is potentially beneficial in unimodal distribution under limited budgets due to finding samples with high volumes.

D.3. Flatness Analysis on Deep Neural Networks

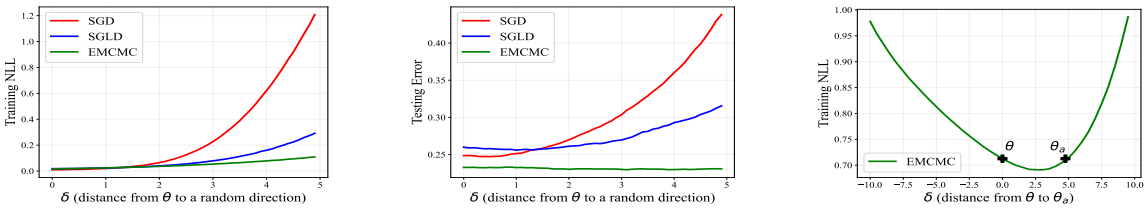
We perform flatness analysis with ResNet18 (He et al., 2016) on CIFAR100 (Krizhevsky, 2009). We use the last samples for SGD, SGLD and EMCMC (averaged result from $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_a$), and each experiment is repeated 3 times to report the averaged scores. We directly interpolating their neighborhood in the parameter space (Izmailov et al., 2018). Local modes located in flat basins are expected to have larger width and better generalization performances (Keskar et al., 2017; Chaudhari et al., 2019). The interpolation begins at $\boldsymbol{\theta}$ and ends at $\boldsymbol{\theta}_\epsilon$ (a random point near $\boldsymbol{\theta}$ or $\boldsymbol{\theta}_\epsilon = \boldsymbol{\theta}_a$). The interpolated point $\boldsymbol{\theta}_\delta$ is computed by:

$$\boldsymbol{\theta}_\delta = (1 - \delta/\|\boldsymbol{\theta} - \boldsymbol{\theta}_\epsilon\|)\,\boldsymbol{\theta} + (\delta/\|\boldsymbol{\theta} - \boldsymbol{\theta}_\epsilon\|)\,\boldsymbol{\theta}_\epsilon, \tag{9}$$

where δ is the Euclidean distance from $\boldsymbol{\theta}$ to $\boldsymbol{\theta}_\delta$. Fig. 5$(a)$ and 5(b) show the training NLL and testing error respectively. The neighborhood of EMCMC maintains consistently lower NLLs and errors compared with SGD and SGLD, demonstrating that EMCMC samples are from flatter modes. Furthermore, Fig. 5(c) visualizes the interpolation between $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_a$, revealing that both variables essentially converge to the same flat mode while maintaining diversity. This justifies the benefit of collecting both of them as samples to obtain a diverse set of high-performing samples.

D.4. Image Classification

We conduct classification experiments on CIFAR (Krizhevsky, 2009), corrupted CIFAR (Hendrycks and Dietterich, 2019b) and ImageNet (Deng et al., 2009), to compare EMCMC with both



(a) $\boldsymbol{\theta} \to$ Random (b) $\boldsymbol{\theta} \to$ Random (c) $\boldsymbol{\theta} \to \boldsymbol{\theta}_a$

Figure 5: Parameter space interpolation of ResNet18 on CIFAR100. Exploring the neighborhood of local modes from $\boldsymbol{\theta}$ to (a)-(b): a random direction in the parameter space, and (c): $\boldsymbol{\theta}_a$. EMCMC has the lowest and the most flat NLL and error curves. (c) shows that $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_a$ converge to the same flat mode while maintaining diversity.

Table 1: Classification results on CIFAR10 and CIFAR100.

Method	CIFAR10		CIFAR100	
	ACC (%) ↑	NLL ↓	ACC (%) ↑	NLL ↓
SGD	94.87 ± 0.04	0.205 ± 0.015	76.49 ± 0.27	0.935 ± 0.021
Entropy-SGD	95.11 ± 0.09	0.184 ± 0.020	77.45 ± 0.03	0.895 ± 0.009
SAM	95.25 ± 0.12	0.166 ± 0.005	78.41 ± 0.22	0.876 ± 0.007
bSAM	95.53 ± 0.09	0.165 ± 0.002	78.92 ± 0.25	0.870 ± 0.005
SGLD	95.47 ± 0.11	0.167 ± 0.011	78.79 ± 0.35	0.854 ± 0.031
Entropy-SGLD	94.46 ± 0.24	0.194 ± 0.020	77.98 ± 0.39	0.897 ± 0.027
EMCMC	$\mathbf{95.69 \pm 0.06}$	$\mathbf{0.162 \pm 0.002}$	$\mathbf{79.16 \pm 0.07}$	$\mathbf{0.840 \pm 0.004}$

flatness-aware optimization methods (Entropy-SGD (Chaudhari et al., 2019), SAM (Foret et al., 2020) and bSAM (Möllenhoff and Khan, 2022)) and MCMC method (SGLD (Welling and Teh, 2011) and Entropy-SGLD (Dziugaite and Roy, 2018)). We use ResNet18 and ResNet50 (He et al., 2016) for CIFAR and ImageNet respectively. All sampling algorithms collect a total of 16 samples for Bayesian marginalization, and all entries are repeated 3 times to report the mean±std. Table 1&2&3 show the results on the 3 datasets, in which EMCMC significantly outperforms all baselines. The classification results strongly suggest that by sampling from flat basins, Bayesian neural networks can achieve outstanding performances and EMCMC is an effective and efficient method to do so.

The results for corrupted CIFAR (Hendrycks and Dietterich, 2019a) are shown in Table 2 to show the robustness of EMCMC against multiple types of noises. The results are averaged over all noise types, and the severity level refers to the strength of noise added to the original data. EMCMC consistently outperforms all compared baselines across all severity levels, indicating that samples from flat basins are more robust to noise. The results for individual noise types are shown in Fig. 10.

Table 2: Classification results on corrupted CIFAR.

Severity	1	2	3	4	5
SGD	88.43	82.43	76.20	67.93	55.81
SGLD	88.61	82.46	76.49	69.19	56.98
EMCMC	**88.87**	**83.27**	**77.44**	**70.31**	**58.17**

Table 3: Classification results on ImageNet.

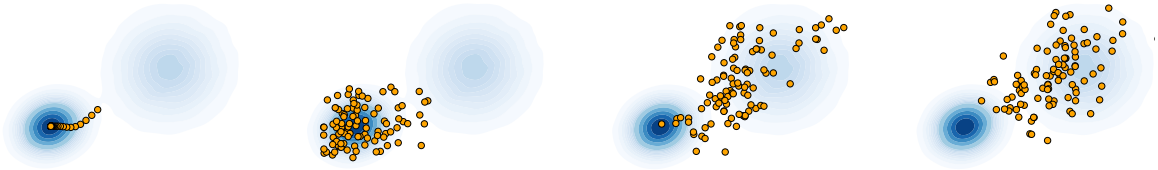
Metric	NLL ↓	Top-1 (%) ↑	Top-5 (%) ↑
SGD	0.960	76.046	92.776
SGLD	0.921	76.676	93.174
EMCMC	**0.895**	**77.096**	**93.424**

D.5. Uncertainty and OOD Detection

To illustrate how predictive uncertainty estimation benefits from flat local geometry, we evaluate EMCMC on out-of-distribution (OOD) detection. We train each model on CIFAR and quantify uncertainty using the entropy of predictive distributions (Malinin and Gales, 2018). Then we use the uncertainty to detect SVHN samples in a joint testing set combined by CIFAR and SVHN (Netzer et al., 2011). We evaluate each algorithm with Area under ROC Curve (AUC) (McClish, 1989) and Expected Calibration Error (ECE) (Naeini et al., 2015). All other settings remain the same as classification experiments. Table 4 shows the evaluation results, where EMCMC outperforms all baselines, especially on the ECE metric. This indicates that predictive uncertainty estimation is more accurate if the posterior samples are from flat basins.

D.6. Additional Synthetic Examples

To demonstrate that EMCMC can bias toward the flat mode under random initialization, we conduct additional synthetic experiments under two different initialization settings. Specifically, we set the initial point to be $(-0.4, -0.4)$ to prefer the sharp mode (Fig. 6) and $(0.0, 0.0)$ to prefer the flat mode (Fig. 7). It is clear that EMCMC can find the flat mode under all initialization settings, while SGD and SGLD are heavily affected by the choices of initialization.

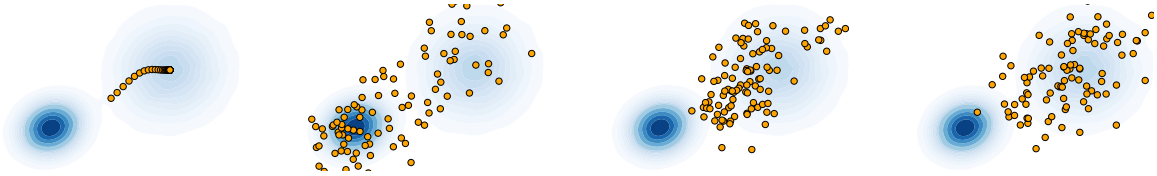


(a) SGD (b) SGLD (c) EMCMC ($\boldsymbol{\theta}$) (d) EMCMC ($\boldsymbol{\theta}_a$)

Figure 6: Synthetic Experiments with sharp-mode-biased initialization.

Table 4: OOD detection on CIFAR-SVHN. The predictive uncertainty quantified by EMCMC is the best among the compared algorithms.

Method	CIFAR10-SVHN		CIFAR100-SVHN	
	AUC (%) ↑	ECE (%) ↓	AUC (%) ↑	ECE (%) ↓
SGD	96.67 ± 0.98	18.09 ± 6.42	74.85 ± 1.69	14.74 ± 2.43
Entropy-SGD	98.17 ± 0.73	6.95 ± 4.22	78.89 ± 2.97	9.30 ± 3.50
SAM	98.01 ± 0.84	3.93 ± 1.19	78.58 ± 1.39	8.16 ± 2.13
bSAM	97.54 ± 0.01	4.31 ± 0.01	79.12 ± 0.01	6.11 ± 0.03
SGLD	97.84 ± 0.26	8.79 ± 1.77	78.18 ± 0.72	9.74 ± 1.55
Entropy-SGLD	95.89 ± 2.64	16.35 ± 8.76	77.50 ± 2.50	6.85 ± 3.57
EMCMC	$\mathbf{98.65 \pm 0.52}$	$\mathbf{2.93 \pm 0.65}$	$\mathbf{79.96 \pm 0.52}$	$\mathbf{4.06 \pm 0.18}$



(a) SGD (b) SGLD (c) EMCMC ($\boldsymbol{\theta}$) (d) EMCMC ($\boldsymbol{\theta}_a$)

Figure 7: Synthetic Experiments with flat-mode-biased initialization.

D.7. Parameter space interpolation

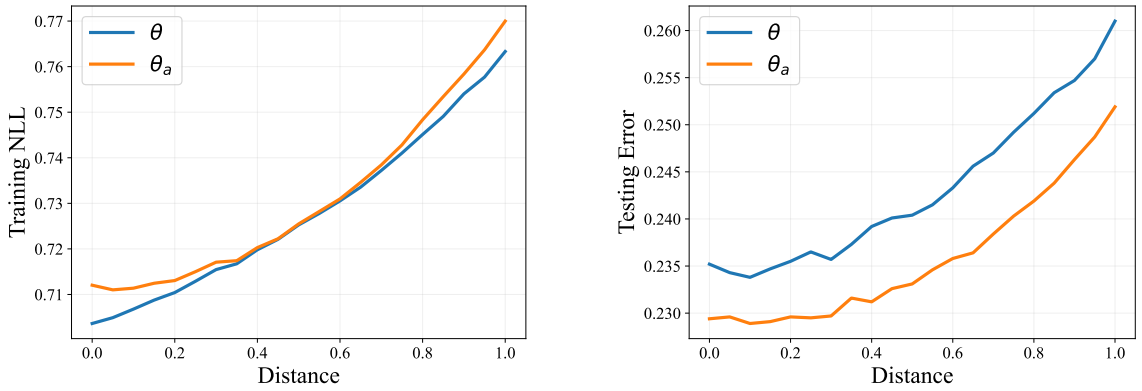
As the supplement for Fig. 5, we show additional interpolation results to demonstrate some interesting findings about the model $\boldsymbol{\theta}$ and the auxiliary guiding variable $\boldsymbol{\theta}_a$. The additional interpolation can be separated into the following types:

D.7.1. toward Random Directions.

We show the interpolation results toward averaged random directions (10 random directions) in Fig. 8. For the training loss, the auxiliary guiding variable $\boldsymbol{\theta}$ is located at a flatter local region with relatively larger loss values. While for the testing error, the guiding variable $\boldsymbol{\theta}_a$ consistently has lower generation errors, which illustrates that the flat modes are more preferable in terms of generalization.

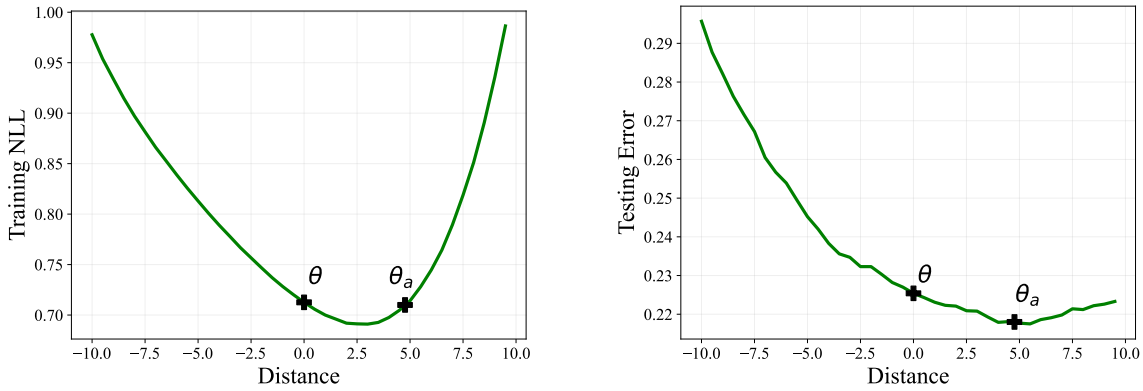
D.7.2. Between Model Parameter and Guiding Variable.

The line between the model parameter $\boldsymbol{\theta}$ and the guiding variable $\boldsymbol{\theta}_a$ is a special direction in the parameter space. The NLL and testing error are both much lower than random directions, which is shown in Fig. 9. Besides, this special direction is biased toward the local region of $\boldsymbol{\theta}_a$, with averagely lower testing errors. This finding justifies the setting of adding $\boldsymbol{\theta}_a$ to the sample set $\mathcal{S}$, since the generalization performance of $\boldsymbol{\theta}_a$ is better.



(a) Random interpolation by training NLL (b) Random interpolation by testing error

Figure 8: Interpolation toward averaged random directions on CIFAR100, comparing the model $\boldsymbol{\theta}$ and the guiding variable $\boldsymbol{\theta}_a$.

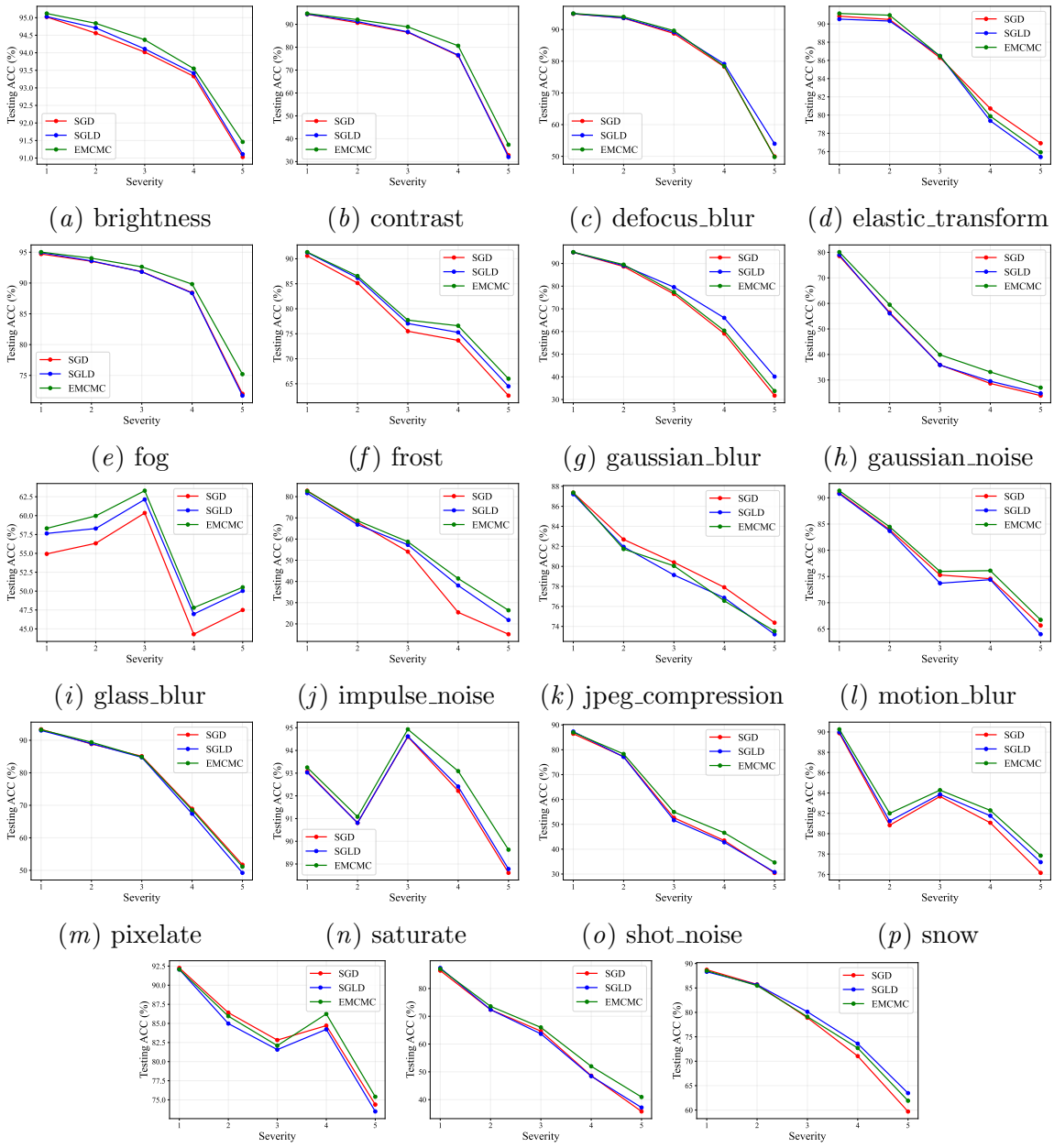


(a) Interpolation between $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_a$ by training NLL (b) Interpolation between $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_a$ by testing error

Figure 9: Interpolation between the model $\boldsymbol{\theta}$ and the guiding variable $\boldsymbol{\theta}_a$ in terms of training NLL and testing error on CIFAR100.

D.8. Classification on Corrupted CIFAR

We list the detailed classification results on corrupted CIFAR (Hendrycks and Dietterich, 2018) in Fig. 10, where each corruption type is evaluated at a corresponding subfigure. For the majority of corruption types, our method ourperforms other baselines under all severity levels, and is superior especially under sever corruptions.



(a) brightness (b) contrast (c) defocus_blur (d) elastic_transform

(e) fog (f) frost (g) gaussian_blur (h) gaussian_noise

(i) glass_blur (j) impulse_noise (k) jpeg_compression (l) motion_blur

(m) pixelate (n) saturate (o) shot_noise (p) snow

(q) spatter (r) speckle_noise (s) zoom_blur

Figure 10: Classification accuracies under different severity levels on corrupted CIFAR. The results are shown per corruption type. Our method outperforms the compared baselines on most of corruption types, especially under high severity levels.

Appendix E. Algorithm Details

We list some details of the proposed Entropy-MCMC in this section, to help understanding our code and reproduction. As discussed in section 2.2, the updating rule of Entropy-MCMC

Algorithm 1: Entropy-MCMC

Inputs: The model $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, guiding variable $\boldsymbol{\theta}_a \in \boldsymbol{\Theta}$, and dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$
Results: Collected samples $\mathcal{S} \subset \boldsymbol{\Theta}$
$\boldsymbol{\theta}_a \leftarrow \boldsymbol{\theta}, \mathcal{S} \leftarrow \emptyset$; `/* Initialize */`
for *each iteration* **do**
  $\boldsymbol{\Xi} \leftarrow$ A mini-batch sampled from $\mathcal{D}$
  $U_{\boldsymbol{\Xi}} \leftarrow -\log p(\boldsymbol{\Xi}|\boldsymbol{\theta}) - \log p(\boldsymbol{\theta}) + \frac{1}{2\eta}\|\boldsymbol{\theta} - \boldsymbol{\theta}_a\|^2$
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha\nabla_{\boldsymbol{\theta}} U_{\boldsymbol{\Xi}} + \sqrt{2\alpha} \cdot \boldsymbol{\epsilon}_1$; `/* ` $\boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_2 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ `*/`
  $\boldsymbol{\theta}_a \leftarrow \boldsymbol{\theta}_a - \alpha\nabla_{\boldsymbol{\theta}_a} U_{\boldsymbol{\Xi}} + \sqrt{2\alpha} \cdot \boldsymbol{\epsilon}_2$
  if *after burn-in* **then**
    $\mathcal{S} \leftarrow \mathcal{S} \cup \{\boldsymbol{\theta}, \boldsymbol{\theta}_a\}$; `/* Collect samples */`
  end
end

can be written as:

$$\widetilde{\boldsymbol{\theta}} \leftarrow \widetilde{\boldsymbol{\theta}} - \alpha\nabla_{\widetilde{\boldsymbol{\theta}}} U(\widetilde{\boldsymbol{\theta}}) + \sqrt{2\alpha} \cdot \boldsymbol{\epsilon}, \tag{10}$$

which is a full-batch version. We will show how to apply modern deep learning techniques like mini-batching and temperature to the updating policy in the following section.

E.1. Mini-batching

We adopt the standard mini-batching technique in our method, which samples a subset of data points per iteration (Li et al., 2014). We assume the entire dataset to be $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$. Then a batch sampled from $\mathcal{D}$ is $\boldsymbol{\Xi} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^M \subset \mathcal{D}$ with $M \ll N$. For the entrie dataset, the loss function is computed by:

$$f(\boldsymbol{\theta}) \propto -\sum_{i=1}^{N} \log p(y_i|\boldsymbol{x}_i, \boldsymbol{\theta}) - \log p(\boldsymbol{\theta}), \tag{11}$$

and in order to balance the updating stride per iteration, the loss function for a mini-batch is:

$$f_{\boldsymbol{\Xi}}(\boldsymbol{\theta}) \propto -\frac{N}{M}\sum_{i=1}^{M} \log p(y_i|\boldsymbol{x}_i, \boldsymbol{\theta}) - \log p(\boldsymbol{\theta}). \tag{12}$$

Therefore, if we average the mini-batch loss over all data points, we can obtain the following form:

$$\bar{f}_{\boldsymbol{\Xi}}(\boldsymbol{\theta}) \propto -\frac{1}{M}\sum_{i=1}^{M} \log p(y_i|\boldsymbol{x}_i, \boldsymbol{\theta}) - \frac{1}{N}\log p(\boldsymbol{\theta}). \tag{13}$$

If we regard the averaging process as a modification on the stepsize α (i.e., $\bar{\alpha} = \alpha/N$), we will have the following form for the updating policy:

$$\begin{aligned}\Delta\widetilde{\boldsymbol{\theta}} &= -\bar{\alpha}\cdot\nabla_{\widetilde{\boldsymbol{\theta}}}\widetilde{U}(\widetilde{\boldsymbol{\theta}}) + \sqrt{2\bar{\alpha}}\cdot\boldsymbol{\epsilon} \\ &= -\bar{\alpha}\cdot\nabla_{\widetilde{\boldsymbol{\theta}}}\left[f_{\boldsymbol{\Xi}}(\boldsymbol{\theta}) + \frac{1}{2\eta}\|\boldsymbol{\theta}-\boldsymbol{\theta}_a\|^2 - \sqrt{\frac{2}{\bar{\alpha}}}\boldsymbol{\epsilon}\odot\widetilde{\boldsymbol{\theta}}\right] \\ &= -\alpha\cdot\nabla_{\widetilde{\boldsymbol{\theta}}}\left[\bar{f}_{\boldsymbol{\Xi}}(\boldsymbol{\theta}) + \frac{1}{2\eta N}\|\boldsymbol{\theta}-\boldsymbol{\theta}_a\|^2 - \sqrt{\frac{2}{\alpha N}}\boldsymbol{\epsilon}\odot\widetilde{\boldsymbol{\theta}}\right].\end{aligned} \tag{14}$$

Therefore, the updating rule in Eq.10 can be equivalently written as:

$$\widetilde{\boldsymbol{\theta}} \leftarrow \widetilde{\boldsymbol{\theta}} - \Delta\widetilde{\boldsymbol{\theta}} \tag{15}$$

E.2. Data Augmentation and Temperature

We apply data augmentation, which is commonly used in deep neural networks, and compare all methods with data augmentation in the main text. Here, we additionally compare the classification results without data augmentation in Table 5 to demonstrate the effectiveness of EMCMC in this case.

Table 5: Comparison of data augmentation of 3 baselines on CIFAR10. EMCMC outperforms previous methods with and without data augmentation.

Augmentation	SGD	SGLD	EMCMC
×	89.60	89.24	**89.87**
✓	95.59	95.64	**95.79**

Besides, in the updating policy, a noise term is introduced to add randomness to the sampling process. However, in mini-batch training, the effect of noise will be amplified so that the stationary distribution of might be far away from the true posterior distribution (Zhang et al., 2020b). Therefore, we also introduce a system temperature T to address this problem.

Formally, the posterior distribution is tempered to be $p(\boldsymbol{\theta}|\mathcal{D}) \propto \exp(-U(\boldsymbol{\theta})/T)$, with an averagely sharpened energy landscape. Similarly, we can regard the temperature effect as a new stepsize $\alpha_T = \alpha/T$, and the updating policy would be:

$$\begin{aligned}\Delta\widetilde{\boldsymbol{\theta}} &= -\alpha\cdot\nabla_{\widetilde{\boldsymbol{\theta}}}\left[\left(\bar{f}_{\boldsymbol{\Xi}}(\boldsymbol{\theta}) + \frac{1}{2\eta N}\|\boldsymbol{\theta}-\boldsymbol{\theta}_a\|^2\right)/T - \sqrt{\frac{2}{\alpha N}}\boldsymbol{\epsilon}\odot\widetilde{\boldsymbol{\theta}}\right] \\ &= -\alpha_T\cdot\nabla_{\widetilde{\boldsymbol{\theta}}}\left[\bar{f}_{\boldsymbol{\Xi}}(\boldsymbol{\theta}) + \frac{1}{2\eta N}\|\boldsymbol{\theta}-\boldsymbol{\theta}_a\|^2 - \sqrt{\frac{2T}{\alpha_T N}}\boldsymbol{\epsilon}\odot\widetilde{\boldsymbol{\theta}}\right].\end{aligned} \tag{16}$$

In order to empirically determine the best temperature, we compare different temperature level in table 6, and find that $T = 10^{-4}$ is appropriate for classification task.

Table 6: Comparison of temperature effect on CIFAR10 with data augmentation. $T = 10^{-4}$ is best temperature level.

T	1	1e-1	1e-2	1e-3	1e-4	1e-5
Testing ACC (%) ↑	95.30	95.42	95.41	95.48	**95.50**	95.47

E.3. Gibbs-like Updating Procedure

Instead of jointly updating, we can also choose to alternatively update $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_a$. The conditional distribution for the model $\boldsymbol{\theta}$ is:

$$p(\boldsymbol{\theta}|\boldsymbol{\theta}_a, \mathcal{D}) = \frac{p(\boldsymbol{\theta}, \boldsymbol{\theta}_a|\mathcal{D})}{p(\boldsymbol{\theta}_a|\mathcal{D})} \propto \frac{1}{Z_{\boldsymbol{\theta}_a}} \exp\left\{-f(\boldsymbol{\theta}) - \frac{1}{2\eta}\|\boldsymbol{\theta} - \boldsymbol{\theta}_a\|^2\right\}, \tag{17}$$

where $Z_{\boldsymbol{\theta}_a} = \exp \mathcal{F}(\boldsymbol{\theta}_a; \eta)$ is a constant. While for the guiding variable $\boldsymbol{\theta}_a$, its conditional distribution is:

$$p(\boldsymbol{\theta}_a|\boldsymbol{\theta}, \mathcal{D}) = \frac{p(\boldsymbol{\theta}, \boldsymbol{\theta}_a|\mathcal{D})}{p(\boldsymbol{\theta}|\mathcal{D})} \propto \frac{1}{Z_{\boldsymbol{\theta}}} \exp\left\{-\frac{1}{2\eta}\|\boldsymbol{\theta} - \boldsymbol{\theta}_a\|^2\right\}, \tag{18}$$

where $Z_{\boldsymbol{\theta}} = \exp\left(-f(\boldsymbol{\theta})\right)$ is a constant. Therefore, with Guassian noise, $\boldsymbol{\theta}_a$ is equivalently sampled from $\mathcal{N}(\boldsymbol{\theta}, \eta \boldsymbol{I})$, and the variance η controls the expected distance between $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_a$. To obtain samples from the joint distribution, we can sample from $p(\boldsymbol{\theta}|\boldsymbol{\theta}_a, \mathcal{D})$ and $p(\boldsymbol{\theta}_a|\boldsymbol{\theta}, \mathcal{D})$ alternatively. The advantage of doing Gibbs-like updating is that sampling $\boldsymbol{\theta}_a$ can be done exactly. Empirically, we observe that joint updating yields superior performance compared to Gibbs-like updating due to the efficiency of updating both $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_a$ at the same time.

Appendix F. Proof of Theorems

F.1. Lemma 1

Proof. Assume $\widetilde{\boldsymbol{\theta}} = [\boldsymbol{\theta}^T, \boldsymbol{\theta}_a^T]^T$ is sampled from the joint posterior distribution:

$$p(\widetilde{\boldsymbol{\theta}}|\mathcal{D}) = p(\boldsymbol{\theta}, \boldsymbol{\theta}_a|\mathcal{D}) \propto \exp\left\{-f(\boldsymbol{\theta}) - \frac{1}{2\eta}\|\boldsymbol{\theta} - \boldsymbol{\theta}_a\|^2\right\}. \tag{19}$$

Then the marginal distribution respectively for $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_a$ would be:

$$\begin{aligned} p(\boldsymbol{\theta}|\mathcal{D}) &= \int_{\boldsymbol{\Theta}} p(\boldsymbol{\theta}, \boldsymbol{\theta}_a|\mathcal{D}) d\boldsymbol{\theta}_a \\ &= \int_{\boldsymbol{\Theta}} \exp\left\{-f(\boldsymbol{\theta}) - \frac{1}{2\eta}\|\boldsymbol{\theta} - \boldsymbol{\theta}_a\|^2\right\} d\boldsymbol{\theta}_a \\ &= \exp(-f(\boldsymbol{\theta})) \int_{\boldsymbol{\Theta}} \exp\left\{-\frac{1}{2\eta}\|\boldsymbol{\theta} - \boldsymbol{\theta}_a\|^2\right\} d\boldsymbol{\theta}_a \\ &= \exp(-f(\boldsymbol{\theta})), \end{aligned} \tag{20}$$

and similarly we have

$$\begin{aligned} p(\boldsymbol{\theta}_a|\mathcal{D}) &= \int_{\boldsymbol{\Theta}} p(\boldsymbol{\theta}, \boldsymbol{\theta}_a|\mathcal{D}) d\boldsymbol{\theta} \\ &= \int_{\boldsymbol{\Theta}} \exp\left\{-f(\boldsymbol{\theta}) - \frac{1}{2\eta}\|\boldsymbol{\theta} - \boldsymbol{\theta}_a\|^2\right\} d\boldsymbol{\theta} \\ &= \exp \mathcal{F}(\boldsymbol{\theta}_a; \eta). \end{aligned} \tag{21}$$

F.2. Lemma 2

Proof Note that we have

$$-\nabla^2 \log \pi_{\text{joint}} = \begin{bmatrix} \nabla^2 f(\theta') + \frac{1}{\eta} I & -\frac{1}{\eta} I \\ -\frac{1}{\eta} I & \frac{1}{\eta} I \end{bmatrix},$$

and after a row reduction, we get

$$\begin{bmatrix} \nabla^2 f(\theta') & 0 \\ -\frac{1}{\eta} I & \frac{1}{\eta} I \end{bmatrix}.$$

The eigenvalues for this matrix are the eigenvalues of $\nabla^2 f(\theta')$ and $1/\eta$. By the assumption $m \leq 1/\eta \leq M$, we have

$$mI \preceq \begin{bmatrix} \nabla^2 f(\theta') & 0 \\ -\frac{1}{\eta} I & \frac{1}{\eta} I \end{bmatrix} \preceq MI,$$

which means $-\nabla^2 \log \pi_{\text{joint}}$ is also a M-smooth and m-strongly convex function.

■

F.3. Proof of Theorem 3

Proof The proof relies on Theorem 4 from Dalalyan and Karagulyan (2019). Lemma 2 has already provided us with the smoothness and strong convexity parameters for π_{joint}. We will now address the bias and variance of stochastic gradient estimation. The stochastic gradient is given by $[\nabla \tilde{f}(\theta') + \frac{1}{\eta}(\theta' - \theta), -\frac{1}{\eta}(\theta' - \theta)]^T$. As $\nabla \tilde{f}(\theta')$ is unbiased and has a variance of σ^2, the stochastic gradient in our method is also unbiased and has variance σ^2. Combining the above results, we are ready to apply Theorem 4 (Dalalyan and Karagulyan, 2019) and obtain

$$W_2(\mu_K, \pi_{\text{joint}}) \leq (1 - \alpha m)^K W_2(\mu_0, \pi) + 1.65(M/m)(2\alpha d)^{1/2} + \frac{\sigma^2 (2\alpha d)^{1/2}}{1.65M + \sigma\sqrt{m}}.$$

■

F.4. Theorem 5

Proof Let $\pi'(\theta') \propto \exp(-f(\theta') - \frac{1}{2\eta}\|\theta' - \theta\|_2^2)$. It is easy to see that $m + 1/\eta \preceq \nabla^2(-\log \pi') \preceq M + 1/\eta$. Based on Theorem 4 in Dalalyan and Karagulyan (2019), the 2-Wasserstein distance for the inner Markov chain is

$$\begin{aligned} W_2(\zeta_L, \pi') &\leq (1-\alpha m)^L W_2(\zeta_0, \pi') + 1.65\left(\frac{M+1/\eta}{m+1/\eta}\right)(\alpha d)^{1/2} + \frac{\sigma^2(\alpha d)^{1/2}}{1.65(M+1/\eta) + \sigma\sqrt{m+1/\eta}} \\ &\leq (1-\alpha m)^L \kappa + 1.65\left(\frac{M+1/\eta}{m+1/\eta}\right)(\alpha d)^{1/2} + \frac{\sigma^2(\alpha d)^{1/2}}{1.65(M+1/\eta) + \sigma\sqrt{m+1/\eta}} \\ &:= A^2. \end{aligned}$$

Now we consider the convergence of the outer Markov chain. We denote $\pi_{\text{flat}}(\theta) \propto \exp \mathcal{F}(\boldsymbol{\theta}; \eta)$. From Chaudhari et al. (2019), we know that

$$\inf_\theta \|\frac{1}{I + \eta\nabla^2 f(\theta)}\| mI \preceq -\nabla^2 \log \pi_{\text{flat}} \preceq \sup_\theta \|\frac{1}{I + \eta\nabla^2 f(\theta)}\| MI.$$

Since $mI \preceq \nabla^2 f(\theta) \preceq MI$, it follows

$$\inf_\theta \|\frac{1}{I + \eta\nabla^2 f(\theta)}\| \geq \frac{1}{1+\eta M}, \qquad \sup_\theta \|\frac{1}{I + \eta\nabla^2 f(\theta)}\| \leq \frac{1}{1+\eta m}.$$

Therefore,

$$\frac{m}{1+\eta M} I \preceq -\nabla^2 \log \pi_{\text{flat}} \preceq \frac{M}{1+\eta m} I.$$

The update rule of the outer SGLD is

$$\theta = \theta - \alpha/\eta(\theta - \mathbf{E}_{\zeta_L}[\theta']) + \sqrt{2\alpha}\xi.$$

The gradient estimation can be written as $\theta - \mathbf{E}_{\pi'}[\theta'] + (\mathbf{E}_{\pi'}[\theta'] - \mathbf{E}_{\zeta_L}[\theta'])$ which can be regarded as the true gradient $\theta - \mathbf{E}_{\pi'}[\theta']$ plus some noise $(\mathbf{E}_{\pi'}[\theta'] - \mathbf{E}_{\zeta_L}[\theta'])$. The bias of the noise can be bounded as follows

$$\begin{aligned} \|\mathbf{E}_{\pi'}[\theta'] - \mathbf{E}_{\zeta_L}[\theta']\|_2^2 &= \| \int [\theta'_{\pi'} - \theta'_{\zeta_L}] dJ(\theta'_{\pi'}, \theta'_{\zeta_L})\|_2^2 \\ &\leq \int \|\theta'_{\pi'} - \theta'_{\zeta_L}]\|_2^2 dJ(\theta'_{\pi'}, \theta'_{\zeta_L}). \end{aligned} \tag{22}$$

Since the inequality holds for any J, we can take the infimum over all possible distributions to conclude

$$\|\mathbf{E}_{\pi'}[\theta'] - \mathbf{E}_{\zeta_L}[\theta']\|_2^2 \leq W_2(\zeta_L, \pi'). \tag{23}$$

Furthermore, we note that the variance of the noise is zero. Therefore, by applying Theorem 4 in Dalalyan and Karagulyan (2019) we get

$$W_2(\nu_K, \pi_{\text{flat}}) \leq (1-\alpha m)^K W_2(\nu_0, \pi_{\text{flat}}) + 1.65\left(\frac{1+\eta M}{1+\eta m}\right)(M/m)(\alpha d)^{1/2} + \frac{A(1+\eta M)}{m}.$$

■

F.5. Theorem 4

Proof Compared to Entropy-SGLD, the only difference of Entropy-SGD is to do SGD update instead of SGLD update in the outer loop. Therefore, the analysis for the inner Markov chain remains the same as in Theorem 5. To analyze the error of SGD in the outer loop, we follow the results in Ajalloeian and Stich (2020). Since the strongly convex parameter for f_{flat} is $\frac{m}{1+\eta M}$, by Section 4.2 and Assumption 4 in Ajalloeian and Stich (2020), we know that

$$\begin{aligned} \frac{1}{2}\|\nabla f_{\text{flat}}(\boldsymbol{\theta}_t)\|^2 &\leq \frac{E_t - E_{t+1}}{\alpha} + \frac{1}{2}A \\ \Rightarrow \frac{m}{1+\eta M}E_t &\leq \frac{E_t - E_{t+1}}{\alpha} + \frac{1}{2}A \\ \Rightarrow E_{t+1} &\leq (1 - \frac{\alpha m}{1+\eta M})E_t + \frac{1}{2}\alpha A. \end{aligned} \tag{24}$$

By unrolling the recursion, we obtain

$$E_K \leq \left(1 - \frac{\alpha m}{1+\eta M}\right)^K E_0 + \frac{A(1+\eta M)}{2m}.$$

■

Appendix G. Ablation Studies

We empirically discuss several important hyper-parameters and algorithm settings in this section, which justifies our choice of their values.

G.1. Variance Term

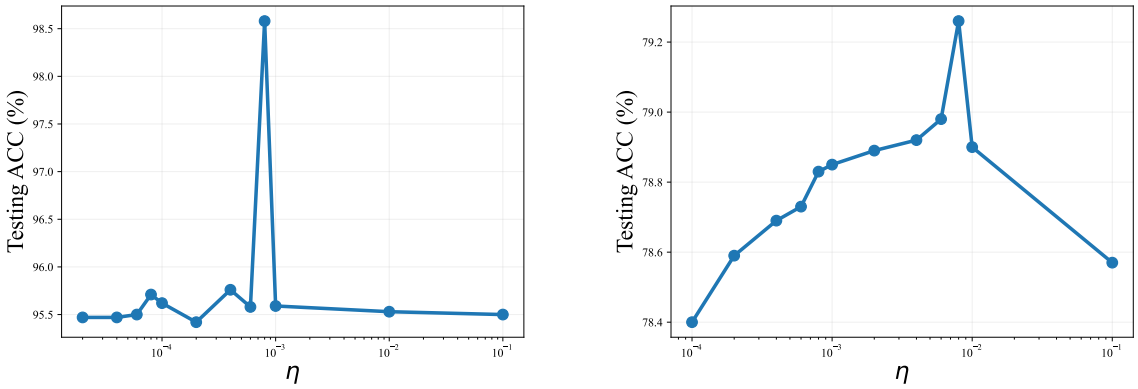
We compare different choices of the variance term η to determine the best value for each dataset. The experimental results are shown in Fig. 11. Generally, the best η for CIFAR100 is about 10^{-2} and for CIFAR10 is about 10^{-3}, which implies that the energy landscapes of CIFAR10 and CIFAR100 may be different.

G.2. Step size Schedules

We compare different types of stepsize schedules in Table 8. Specifically, we assume the initial and final stepsize to be α_0 and α_1 respectively. T is the total number of epochs and t is the current epoch. The detailed descriptions of stepsize schedules are listed in Table 7. The cyclical stepsize is the best among all stepsize schedules.

G.3. Collecting Samples

Due to the introduction of the auxiliary guiding variable $\boldsymbol{\theta}_a$, the composition of sample set $\mathcal{S}$ has multiple choices: only collect samples of $\boldsymbol{\theta}$, only collect samples of $\boldsymbol{\theta}_a$, collect both samples. We conduct the comparison of all choices and the results are reported in Table 9. It shows that using samples from both $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_a$ gives the best generalization accuracy.



(a) CIFAR10 (b) CIFAR100

Figure 11: Comparison of different variance level for Entropy algorithm. $\eta \approx 10^{-3}$ is the best on CIFAR10, and $\eta \approx 10^{-2}$ is the best on CIFAR100.

Table 7: Formulas or descriptions of different stepsize schedules.

Name	Formula/Description
constant	$\alpha(t) = \alpha_0$
linear	$\alpha(t) = \frac{T-t}{T}(\alpha_0 - \alpha_1) + \alpha_1$
exponential	$\alpha(t) = \alpha_0 \cdot (\alpha_1/\alpha_0)^{t/T}$
step	Remain the same stepsize within one "step", and decay between "steps".
cyclical	Follow Eq. 1 in Zhang et al. (2020b).

G.4. Normalization Layers

During testing, the usage of bath normalization layers (BN) in the model architecture induces a problem regarding the mini-batch statistics. The mean and variance of a batch need calculated through at least one forward pass, which is not applicable for the guiding variable $\boldsymbol{\theta}_a$ since it is updated by the distance regularization during training. We try different solutions for this problem, including one additional forward pass and the Filter Response Normalization (Singh and Krishnan, 2020). The comparison is listed in Table 10, where simply adding one additional forward pass during testing can achieve promising accuracy with negligible computational overhead.

G.5. SGD Burn-in

We also try SGD burn-in in our ablation studies, by adding the random noise term only to the last few epochs to ensure the fast convergence. We evaluate different settings of SGD burn-in epochs in Table 11. We find that adding 40 burn-in epochs per 50 epochs is the best choice.

Table 8: Comparison of stepsize schedules on CIFAR100. The cyclical stepsize is the best for Entropy-MCMC.

Learning Rate Schedule	constant	linear	exponential	step	cyclical
Testing ACC (%) ↑	88.04	87.89	87.75	89.59	**89.93**

Table 9: Ablation study on the composition of sample set $\mathcal{S}$ on CIFAR10. With samples from both $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_a$, the Bayesian marginalization can achieve the best accuracy.

$\boldsymbol{\theta}$	$\boldsymbol{\theta}_a$	ACC (%) ↑
✓		95.58
	✓	95.64
✓	✓	**95.65**

Appendix H. Conclusion and Discussion

We propose a practical MCMC algorithm to sample from flat basins of DNN posterior distributions. Specifically, we introduce a guiding variable based on the local entropy to steer the MCMC sampler toward flat basins. We prove the fast convergence rate of our method compared with two existing flatness-aware methods. Comprehensive experiments demonstrate the superiority of our method, verifying that it can sample from flat basins and achieve outstanding performances on diverse tasks. Our method is mathematically simple and computationally efficient, allowing for adoption as a drop-in replacement for standard sampling methods such as SGLD.

The results hold promise for both Bayesian inference and deep learning generalization. On the one hand, we demonstrate that explicitly considering flatness in Bayesian inference can significantly enhance the generalization, especially under practical computational constraints. On the other hand, we highlight the value of marginalizing over flat basins in the energy landscape, as a means to attain further improvements in generalization compared to single point optimization.

Table 10: Comparison of different normalization layers on CIFAR10. Simply adding one additional forward pass during testing with standard batch normalization is the best solution.

Normalization Layer	ACC (%) ↑	Time (h)
BN	95.40	**1.8**
BN (one additional forward)	**95.47**	1.9
FRN (Singh and Krishnan, 2020)	93.92	2.5

Table 11: Comparison of different SGD burn-in epochs on CIFAR10. In a 50-epoch round, using SGD burn-in in the first 40 epochs is the best choice.

SGD Burn-in Epoch	0	10	20	30	40	47
Test ACC (%) ↑	95.61	95.62	95.57	95.67	**95.72**	95.41