BAYESIAN DEEP EQUILIBRIUM MODELS WITH SEQUENTIAL INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep Equilibrium Models (DEQs) have drawn considerable attention due to their unique advantages. However, their uncertainty estimation, crucial for prediction-sensitive applications, such as prediction in scientific research, remains unexplored. In this paper, we propose Bayesian Deep Equilibrium Models to address this gap for the first time. Our study highlights the substantial computational cost associated with uncertainty estimation in Bayesian DEQs. To mitigate this challenge, we introduce a novel sequential inference approach that captures the similarities in the parameters and reduces computational redundancy in the inference, offering a promising method to accelerate uncertainty quantification in DEQs. We also provide theoretical justification for the motivation behind our approach. Comprehensive experiments on Darcy Flow, Navier-Strokes equations, and computer vision datasets demonstrate that our method can speed up uncertainty estimation with Bayesian DEQs by up to 3 times without any sacrifice in performance.

1 INTRODUCTION

Deep equilibrium models (DEQs) (Bai et al., 2019) have attracted considerable attention in recent years due to their broad range of applications (Bai et al., 2022; Gu et al., 2020; Bai et al., 2020). Unlike traditional deep neural networks, which define representations via a finite composition of layers, DEQs use a fixed-point formulation to implicitly define the underlying representation. In this sense, a DEQ can be interpreted as a weight-tied neural network with an infinite number of layers. Leveraging this formulation, DEQs often outperform traditional architectures on problems with an inherent fixed-point structure, such as solving differential equations and modeling physical systems (Marwah et al., 2023). The implicit nature of DEQs also leads to distinctive statistical properties, including robustness (Liu et al., 2021). Beyond these modeling benefits, DEQs enjoy several practical advantages: (i) they are memory-efficient during training, since gradients are computed via the implicit function theorem rather than backpropagating through a stored computation graph (Bai et al., 2019); and (ii) their optimization framework naturally exposes an accuracy–efficiency trade-off, making them suitable for fast-iterative inference scenarios (Chen et al., 2018). Together, these characteristics highlight the growing importance of DEQs in modern deep learning.

Quantifying model uncertainty is essential in scientific research, where learned models approximate complex physical systems and simulations. Reliable uncertainty estimates are important when predictions are made under poorly represented physical conditions or parameter regimes (Abdar et al., 2021). This necessitates the development of trustworthy deep models that can indicate the reliability of their predictions. Furthermore, uncertainty estimation is beneficial in tasks such as out-of-distribution detection and active learning (Yang et al., 2024; Settles, 2009). In this context, Bayesian Neural Networks (BNNs) have been proposed as a powerful solution for uncertainty quantification and generalization ability (MacKay, 1992). BNNs model the parameters as a posterior distribution rather than a point estimation, thereby enabling uncertainty estimation. Approaches have been introduced to approximate the posterior, including Monte Carlo Markov Chain (MCMC) (Welling & Teh, 2011), and Stochastic Weight Averaging-Gaussian (SWAG) (Maddox et al., 2019).

A growing body of research has focused on various aspects of DEQs, such as their accuracy, robustness, and efficiency (Bai et al., 2020; Yang et al., 2022; Bai et al., 2021a). However, the uncertainty associated with DEQs remains largely unexplored, raising concerns about the trustworthiness of their application. Since DEQs are typically trained using stochastic gradient descent (SGD) (Robbins & Monro, 1951), they only provide point estimates for predictions, making it challenging to

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

compute uncertainty measurements such as negative log-likelihood (NLL) in regression tasks. Moreover, our preliminary study (Figure 1) reveals that DEQs suffer from poor calibration in classification tasks, similar to traditional neural networks (Guo et al., 2017), highlighting the uncertainty estimation issue for DEQs, especially in sensitive applications.

In this paper, we present the first comprehensive study on Bayesian Deep Equilibrium Models to address the uncertainty estimation problem in DEQs. Our findings indicate that incorporating MCMC and SWAG into DEQs achieves strong uncertainty estimation. However, it incurs substantial computational costs due to the need for multiple inferences. For instance, MCMC requires at least 10 samples to approximate the posterior distribution, resulting in approximately 80 seconds to estimate the uncertainty for solving a steady-state PDE equation (Darcy Flow) on a single A6000 GPU. This computational burden poses a significant challenge for deploying Bayesian DEQs in sensitive scenarios. In this study, we propose a novel sequential sampling to capture the similarities of the samples in the posterior distribution and further introduce sequential computation to eliminate the redundancy in the inference of DEQs. We integrate these two techniques into a unified framework, termed sequential inference, which significantly accelerates Bayesian DEQs without sacrificing performance. In summary, our contributions are as follows:

- We provide the first study to investigate uncertainty estimation of DEQs, showing that point-wise estimation leads to bad uncertainty evaluation in regression and classification.
- We introduce Bayesian DEQs to estimate uncertainty, but uncover the computational issue and further reveal the redundancy during the inference.
- We propose sequential inference to improve the efficiency of Bayesian DEQs and provide a theoretical justification to clarify the rationale of our design.
- We conduct extensive experiments on solving differential equations and image classification to validate the effectiveness of our approach, demonstrating that our method can accelerate the inference speed of Bayesian DEQs by up to $\mathbf{3}\times$ without sacrificing performance.

2 Related Works

2.1 Deep Equilibrium Models

Recently, there has been a surge of research on deep implicit models defining outputs through implicit functions (Amos & Kolter, 2017; Chen et al., 2018; Bai et al., 2019; Agrawal et al., 2019; El Ghaoui et al., 2021). Among these, deep equilibrium models define implicit layers by solving a fixed-point problem (Bai et al., 2019; 2020). DEQs achieve state-of-the-art performance on tasks that benefit from implicit modeling, such as solving differential equations (Marwah et al., 2023), and perform on par with deep neural networks across applications, including image recognition (Bai et al., 2020), image generation (Pokle et al., 2022), graph modeling (Gu et al., 2020; Chen et al., 2022), and equation solving (Marwah et al., 2023). Though DEQs are competitive in terms of performance, the computational inefficiency remains a bottleneck due to their iterative nature.

2.2 Bayesian Neural Networks

Deep neural networks for classification are often prone to being overconfident (Guo et al., 2017) and provide unreliable uncertainty estimation (Abdar et al., 2021). Moreover, the point estimation nature of deep neural networks cannot provide uncertainty estimation in regression tasks without the variance information. Bayesian neural networks (BNNs) have been proposed to address this issue by estimating the posterior distribution of the parameters (MacKay, 1992). Monte Carlo Markov Chain (MCMC) constructs Markov chains where the samples generated follow the posterior distribution (Bardenet et al., 2017). Several approaches have adapted MCMC to stochastic gradient descent (Welling & Teh, 2011; Chen et al., 2014). Stochastic Weight Averaging-Gaussian (SWAG) estimates the posterior using a Gaussian distribution (Maddox et al., 2019) by collecting parameters along the training trajectory of SWA (Izmailov et al., 2018), making it compatible with modern training practices for DNNs. Laplacian Approximation (LA)(Daxberger et al., 2021) approximates the posterior distribution with a Gaussian by leveraging the Hessian matrix. However, such methods either require multiple forward passes during inference, resulting in significant computational overhead, or not suitable for DEQs, as discussed in Appendix D.

Another line of research focuses on uncertainty estimation using a single forward pass, which typically modify the final layer to facilitate uncertainty estimation for efficiency (Liu et al., 2020; Van Amersfoort et al., 2020). However, Bayesian-based methods remain dominant due to their distinct advantages. First, single-pass methods often yield less reliable uncertainty estimation, particularly on large-scale datasets (Abdar et al., 2021). Second, their deterministic nature makes them prone to overfitting and sensitive to the training and hyperparameters, requiring substantial data to achieve robust performance (Gawlikowski et al., 2023). The limitations underscore the importance of research into ensemble-based uncertainty estimation techniques, which is the focus of this paper.

3 Bayesian Deep Equilibrium Models

In this section, we begin with preliminary studies to demonstrate that uncertainty estimation is a challenging task for DEQs. To tackle this issue, we incorporate Bayesian inference methods into DEQs and investigate the computational cost challenges in practical settings.

3.1 Deep Equilibrium Models

Explicit neural networks typically perform the forward pass using explicit functions, represented as $\boldsymbol{z}^{l+1} = f^l(\boldsymbol{z}^l; \theta_l)$, where $\boldsymbol{z}^l$ and $\boldsymbol{z}^{l+1}$ denote the input and output of the layer $f^l(\cdot; \theta_l)$, parameterized by θ_l. In contrast, DEQs define their outputs by a fixed-point problem: $\boldsymbol{z}^* = f(\boldsymbol{z}^*, \boldsymbol{x}; \theta)$, where $\boldsymbol{z}^*$ is the output representation and $\boldsymbol{x}$ is the input data. Consequently, DEQs require solving a fixed-point equation for each input $\boldsymbol{x}$ to compute the representation $\boldsymbol{z}^*$. Several fixed-point solvers have been employed for DEQs, including fixed-point iteration, Anderson acceleration, and Broyden's method (Bai et al., 2019). For instance, fixed-point iteration updates the solution repeatedly until the representation $\boldsymbol{z}$ converges: $\boldsymbol{z}^{l+1} = f(\boldsymbol{z}^l, \boldsymbol{x}; \theta)$. In this paper, we represent the fixed-point solver as:

$$\boldsymbol{z} = \text{Solver}(f, \boldsymbol{x}, \boldsymbol{z}^0; \theta), \tag{1}$$

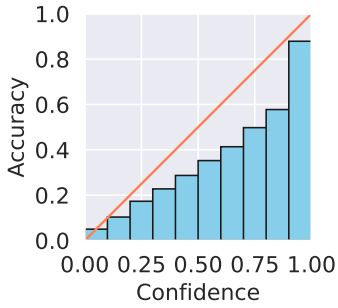


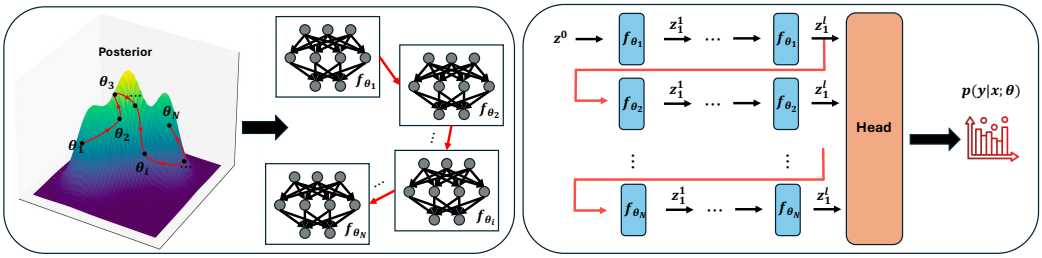
Figure 1: Reliability diagram for MDEQ trained with SGLD on CIFAR-10. The confidence in each bin is higher than the corresponding accuracy, showing that the model is overconfident.

where $\boldsymbol{z}^0$ is the initial guess. Although deep learning achieves high accuracy, its uncertainty estimation remains a significant concern, particularly in sensitive applications. This issue occurs in DEQs as well. First, DEQs trained using SGD only provide point estimates for predictions, limiting their ability to compute uncertainty measurements in regression tasks (Jospin et al., 2022). For example, the computation of the negative log-likelihood (NLL) commonly assumes a Gaussian predictive distribution; however, a point estimate offers no information about the predictive variance, rendering the NLL ill-defined. Moreover, similar to traditional neural networks, DEQs also exhibit overconfidence in classification tasks, where overconfidence means the confidence of predictions is higher than the accuracy (Guo et al., 2017). To illustrate this, we evaluate the calibration of MDEQ on CIFAR-10 and plot a reliability diagram comparing the confidence of the model to its accuracy, where MDEQ is a particular implementation of DEQs designed for computer vision tasks, adhering to all the general properties of DEQs. As shown in Figure 1, the confidence in each bin exceeds its accuracy, highlighting a significant overconfidence issue in DEQs.

3.2 Uncertainty Estimation

BNNs (MacKay, 1992) provide a valid solution to address the above challenge. BNNs treat parameters as distributions rather than point estimates, allowing the final predictions to capture uncertainty through statistical information. In this section, we discuss methods for approximating the posterior in large-scale models and analyze the challenges of applying these approaches to DEQs.

MCMC. To approximate the posterior distribution, we can employ the Monte Carlo Markov Chain (MCMC) method, which iteratively updates the model distribution through a Markov chain until it converges to the posterior distribution. To facilitate batch-wise training in deep learning, techniques such as Stochastic Gradient Langevin Dynamics (SGLD) have been introduced (Welling & Teh,



(a) Sequential Sampling (b) Sequential Computation

Figure 2: Illustrations of our proposed sequential inference. (a) *Step 1*: sequential sampling draws parameters from the posterior in a sequential manner, capturing parameter similarities. (b) *Step 2*: sequential computation initializes with the previous and converges to the fixed point within a few iterations. Finally, we aggregate the predictions to approximate the output distribution.

2011), which integrate Langevin dynamics with mini-batch gradient updates:

$$\Delta\theta_t = \frac{\epsilon_t}{2}\left(\nabla \log p(\theta_t) + \frac{N}{n}\sum_{i=1}^{n} \nabla \log p(\boldsymbol{x}_i|\theta_t)\right) + \eta_t,$$

where $\eta_t \sim N(0, \epsilon_t)$, ϵ_t denotes the learning rate, θ_t denotes the model parameters, $p(\cdot)$ represents the likelihood, n is the batch size, and N is the size of the dataset. Compared to SGD, SGLD introduces additional noise at each iteration. By storing the parameters θ_t in the optimization process, we can approximate the posterior distribution and aggregate the resulting predictions.

SWAG. Stochastic Weight Averaging-Gaussian (SWAG) approximates the posterior using a Gaussian distribution (Maddox et al., 2019). Suppose we train the model for T epochs. Then the mean of the Gaussian distribution is computed as the average of the parameters over these epochs: $\theta_{\text{SWAG}} = \frac{1}{T}\sum_{i=1}^{T}\theta_i$, where θ_i denotes the model parameters at epoch i. To approximate the covariance of the Gaussian distribution, SWAG defines Σ_{Diag} and $\Sigma_{\text{low-rank}}$:

$$\Sigma_{\text{Diag}} = \text{diag}(\overline{\theta^2} - \theta^2_{\text{SWAG}}), \tag{2}$$

$$\Sigma_{\text{low-rank}} = \frac{1}{K-1}\sum_{i=T-K+1}^{K}(\theta_i - \bar{\theta}_i)(\theta_i - \bar{\theta}_i)^\top, \tag{3}$$

where $\overline{\theta^2} = \frac{1}{T}\sum_{i=1}^{T}\theta_i^2$ and $\bar{\theta}_i = \frac{1}{i}\sum_{j=1}^{i}\theta_j$. The "diag" operator denotes a diagonal matrix, and K represents the number of weights used to approximate $\Sigma_{\text{low-rank}}$. $\Sigma_{\text{low-rank}}$ captures the correlations between parameters in a memory-efficient way, while Σ_{Diag} is used to stabilize the covariance matrix. The final covariance matrix is estimated as:

$$\Sigma_{\text{SWAG}} = s(\Sigma_{\text{Diag}} + \Sigma_{\text{low-rank}}), \tag{4}$$

where s is a scaling factor. Note that, in practice, we only store the diagonal elements and $\theta_i - \bar{\theta}$, where $i = T - K + 1, \cdots, T$, instead of directly storing Σ_{SWAG}. In this paper, we adapt these two BNN methods for DEQs by modifying the training settings, as detailed in Section 5.1. However, this adaptation introduces computational challenges, as demonstrated below.

Inference of Bayesian DEQs is computationally expensive. There are three primary factors contributing to the high computational cost. First, DEQs often have complex architectures to maintain expressiveness compared to standard models. For example, MDEQs incorporate multiple convolutional layers within each fixed-point iteration. Second, achieving the fixed point requires numerous iterations using a solver. Although some higher-order solvers reduce the number of iterations, they demand more computation time per step. Finally, uncertainty estimation requires running DEQs multiple times to approximate the posterior distribution. For instance, MCMC requires at least 10 samples to approximate the posterior distribution, resulting in approximately 74 seconds to estimate the uncertainty for solving a steady-state PDE equation (Darcy Flow) on a single A6000 GPU. Collectively, these factors make Bayesian DEQs resource-intensive.

4 Sequential Inference

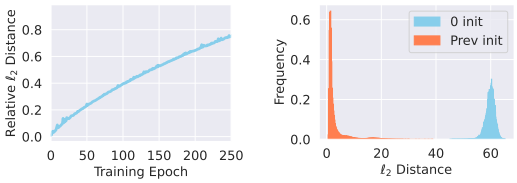
Uncertainty estimation typically requires multiple sampling and inferences, resulting in a computational bottleneck. To address this challenge, we propose sequential inference, which is composed of

two techniques: sequential sampling and sequential computation. In this section, we first introduce the sequential sampling method, which captures the similarities in the parameters while maintaining the posterior distribution. We present how they are incorporated into classic Bayesian methods: MCMC and SWAG. We then present the sequential computation technique, which exploits the similarities to reduce computational redundancy in the forward process.

4.1 Sequential Sampling

Uncertainty estimation methods require multiple inferences using different parameter samples. However, current approaches for ensembling predictions overlook the similarity among the parameters resulting from the posterior distribution. In this section, we propose a sequential sampling method to capture similarities and integrate them into existing Bayesian inference techniques. Our approach aims to "order" the samples such that nearby parameters remain close while preserving their distribution at no additional cost. The proposed sampling strategy is illustrated in Figure 2a.

MCMC. We collect parameters from the posterior using a Markov Chain in MCMC. Due to the nature of gradient descent, consecutive parameter samples tend to be close to each other. As a result, the samples follow an order in which nearby samples remain close. To validate the feasibility of our method, we conduct a preliminary study on Darcy Flow using MDEQ trained with SGLD and compute the relative ℓ_2 distance between the parameters from the first epoch and those from subsequent training epochs. As shown in Figure 3a, the distances between the nearby epochs are smaller than non-nearby sample pairs, indicating that our sequential sampling effectively captures the similarity.



(a) (b)

Figure 3: (a) The relative ℓ_2 distance between the parameters from the first epoch and those from subsequent training epochs in SGLD. (b) The distribution of distances between the initialization and the fixed point with different initializations for SGLD.

SWAG. In SWAG, the samples are drawn from the Gaussian distribution $\mathcal{N}(\theta_{\text{SWAG}}, \Sigma_{\text{SWAG}})$. For memory efficiency, we sample parameters from $\mathcal{N}(\theta_{\text{SWAG}}, \Sigma_{\text{SWAG}})$ using the following approach:

$$\hat{\theta} = \theta_{\text{SWAG}} + \sqrt{s}\Sigma_{\text{Diag}}^{\frac{1}{2}}\xi^{(1)} + \frac{\sqrt{s}}{\sqrt{K-1}}\hat{D}\xi^{(2)} \tag{5}$$

where $\hat{D}$ is the concatenation of $\theta_i - \overline{\theta}_i$. Here, $\xi^{(1)} \sim \mathcal{N}(0, I_d)$ corresponds to the dimension of the model parameters, and $\xi^{(2)} \sim \mathcal{N}(0, I_K)$. According to Eq. (5), we can utilize Langevin dynamics to sample $\xi^{(1)}$ and $\xi^{(2)}$ from the Gaussian distribution to capture the similarity. We sample $\xi \sim \mathcal{N}(0, I_{d+K})$ and split it into $\xi^{(1)}$ and $\xi^{(2)}$. Considering the probability density function of $\mathcal{N}(0, I_{d+K})$, the samples are drawn in the following sequential manner:

$$\xi_{t+1} = (1-\tau)\xi_t + \sqrt{\tau}\eta, \text{ where } \eta \sim \mathcal{N}(0, I_{d+K}) \tag{6}$$

where τ is the step size of Langevin dynamics. Although Langevin dynamics itself converges to the posterior distribution, its Euler discretization in Eq. 6 introduces discretization bias. To ensure that the sampling procedure converges to the target posterior distribution, we apply a Metropolis–Hastings correction (Kalos & Whitlock, 2008) to determine whether each proposed update should be accepted. A detailed theoretical explanation is provided in Appendix L.

Remark 1 *We clarify that the similarity here is different from auto-correlation in Monte Carlo sampling. Our sequential sampling method "orders" the samples while remaining representative of the posterior distribution. We empirically evaluate the representativeness of samples in sequential sampling in Appendix H, where the Wasserstein distance between the samples from sequential sampling and normal sampling is pretty small. We also provide the study of the effective number in Appendix L.*

4.2 Sequential Computation

In this section, we propose sequential computation to reduce the redundancy of similar forward processes caused by similar parameters captured in sequential sampling. The core idea is to leverage

historical information to accelerate subsequent computations. Considering the importance of initialization (Bai et al., 2021a), we utilize the fixed point from the previous iteration as the initialization for the next fixed-point problem, instead of starting from the standard zero initialization. This strategy reduces the initial distance between the starting point and the final fixed-point solution, thereby simplifying the optimization problem. We validate the effectiveness of our initialization by comparing the distance between consecutive fixed points to the distance between a fixed point and zero initialization, following the setting in Section 4.1. As shown in Figure 3b, the previous initialization is significantly closer to the fixed-point solution compared to the standard zero initialization.

To provide an intuitive understanding, consider the fixed-point iteration solver in Eq. (1) as analogous to layers in a neural network, where each iteration represents one layer. Using the previous fixed point as the initialization is similar to starting from an intermediate layer, thereby skipping the computations required for the first many layers and eliminating redundancy. For example, if a neural network has 50 layers and our method initializes from an equivalent state at the 45th layer, this approach can achieve up to $10\times$ acceleration. To be specific, our method updates the parameters as follows:

$$\boldsymbol{z}_i^l = \text{Solver}(f, \boldsymbol{x}, \boldsymbol{z}_{i-1}^l; \theta_i), \tag{7}$$

where $i = 1, 2, \ldots, N$ and l is the number of the fixed-point iteration to approximate z_i^*. We illustrate our method in mini-batch mode on SWAG in Algorithm 1. Other Bayesian methods refer to Appendix C. We demonstrate the feasibility of our approach in the following theorem with proper assumptions:

Algorithm 1 Sequential Inference for SWAG

Require: DEQ $f(\cdot)$, SWAG mean θ_{SWAG}, SWAG diagonal Σ_{Diag}, SWAG correlation $\hat{D}$, SWAG scaling s, Langevin step size τ, input data $\boldsymbol{x}$

Sample Gaussian noise $\xi_0 \in \mathbb{R}^{d+K}$
Fixed-point initialization: $\boldsymbol{z}^0 = \mathbf{0} \in \mathbb{R}^{B\times d}$
Prediction posterior initialization: $\mathbf{p}(\mathbf{y}|\mathbf{x};\theta) = [\ ]$
for $1 \le i \le N$ **do**
  Sample Gaussian noise $\eta \in \mathbb{R}^{d+K}$
  $\xi_{i+1} = (1-\tau)\xi_i + \sqrt{\tau}\eta$
  $\xi^{(1)} = \xi_{i+1}[0, d], \xi^{(2)} = \xi_{i+1}[d+1, d+K]$
  $\theta_i = \theta_{\text{SWAG}} + \sqrt{s}\Sigma_{\text{Diag}}^{1/2}\xi^{(1)} + \frac{\sqrt{s}}{\sqrt{K-1}}\hat{D}\xi^{(2)}$
  $\boldsymbol{z}^i = \text{Solver}(f, \boldsymbol{x}, \boldsymbol{z}^{i-1}; \theta_i)$
  Classify $\boldsymbol{z}^i$ to get $\mathbf{p}(\mathbf{y}|\mathbf{x};\theta_\mathbf{i}) \in \mathbb{R}^C$
  Append $\mathbf{p}(\mathbf{y}|\mathbf{x};\theta_\mathbf{i})$ to $\mathbf{p}(\mathbf{y}|\mathbf{x};\theta)$
end for
Return $\mathbf{p}(\mathbf{y}|\mathbf{x};\theta)$

Theorem 1 *Consider a sequence of parameter samples $\{\theta_1, \theta_2, \ldots, \theta_N\}$ and a DEQ $f(\boldsymbol{z}, \boldsymbol{x}; \theta)$. Suppose the following conditions are satisfied:*

- *f has non-sharp fixed points and is smooth within a local region C, where $\{\theta_1, \theta_2, \ldots, \theta_N\} \subset C$. Specifically, there exist constants $M_1 > 0$ and $M_2 > 0$ such that for all $\theta \in C$,*

$$\left\| \left(I - \frac{\partial f}{\partial z}(\cdot;\theta) \right)^{-1} \right\|_2 \le M_1 \text{ and } \left\| \frac{\partial f}{\partial \theta}(\cdot;\theta) \right\|_2 \le M_2,$$

- *The parameters in the sequence are close to each other: there exists $\delta > 0$ such that for any $0 \le i \le N-1$, $\|\theta_{i+1} - \theta_i\|_2 \le \delta$.*

For any $0 \le i \le N-1$, the following inequality holds: $\|\boldsymbol{z}_{i+1}^* - \boldsymbol{z}_i^*\|_2 \le \sqrt{d}\delta M_1 M_2$, where $\boldsymbol{z}_i^* \in \mathbb{R}^d$ is the solution of fixed-point problems with respect to parameter θ_i.

The proof is shown in Appendix A. The above theorem demonstrates that the distance between the fixed points corresponding to nearby samples is bounded by the smoothness of the DEQ. Thus, using the previous fixed point as initialization offers a better starting point for the following fixed-point problem. The two assumptions generally hold. First, it is common to assume the smoothness of a well-trained DEQ. Since DEQs converge, the change in fixed points is small, implying that M_1, the gradient with respect to $\boldsymbol{z}$, remains small. Additionally, M_2, the gradient with respect to θ, is small, as the model reaches an optimal state during training. Second, the sequential property holds for Bayesian inference methods using our sequential sampling approach. In Appendix J, we empirically approximate the magnitudes of M_1 and M_2, which supports our assumption.

Remark 2 *Our sequential inference is compatible with parallel computing during implementation. Since multi-model inference typically requires multi-process or multi-node distributed computing (Chandra et al., 2019), which is challenging, it is common to use batch-wise evaluation for BNNs. This approach allows us to process a batch of data and apply sequential computation in parallel across the batch, which is the setting applied in our experiments.*

5 EXPERIMENTS

In this section, we first introduce the experimental settings, including the datasets, model architectures, Bayesian inference methods, and evaluation metrics. Next, we apply these Bayesian inference methods to DEQs and demonstrate how our sequential inference technique integrates with them. Finally, we perform a comprehensive analysis of our sequential inference through ablation studies.

5.1 EXPERIMENTAL SETTINGS

Datasets & DEQ Settings. We first evaluate the effectiveness of our method on synthetic regression datasets. We then consider a steady-state PDE task, where the goal is to regress solution values given spatial coordinates and the coefficients of the underlying equation. Specifically, we focus on the Darcy flow and Navier–Stokes equations (Marwah et al., 2023), with dataset details provided in Appendix B.1. For the synthetic datasets, we use a DEQ with fully connected layers (FC-DEQ), following the configuration in MonoDEQ (Winston & Kolter, 2020), while for the steady-state PDEs we adopt FNO-DEQ, which is specifically designed for this class of problems. Additional details about the model configurations are provided in Appendix B.

We emphasize that our method is general and can be applied to any DEQ architecture. To further demonstrate its generality and scalability, we also present experimental results on three popular image classification benchmarks in Appendix D: MNIST (LeCun et al., 1998), CIFAR-10 (Krizhevsky et al., 2009), and ImageNet (Deng et al., 2009) with the use of MDEQ (Bai et al., 2020).

Bayesian Methods. We adopt two Bayesian methods for uncertainty estimation as discussed in Section 3 in the main paper. For MCMC, we choose to use pre-conditioned Stochastic Gradient Langevin Dynamics, which is a variant of SGLD[1]. We use the size of the training dataset to define N in Langevin dynamics (Welling & Teh, 2011). For SWAG, we approximate the covariance matrix following the paper (Maddox et al., 2019). For the notation clarity, we use the Bayesian inference method (e.g., SGLD, SWAG) appended to the model name to denote the training method. For example, FC-DEQ-SGLD refers to the FC-DEQ model trained with SGLD. For more details on the implementation of posterior estimation in Bayesian models, please refer to Appendix B. To further show the generalization of our methods, we conduct experiments on MC-Dropout (Gal & Ghahramani, 2016), Deep ensembles (Lakshminarayanan et al., 2017), Hamiltonian Monte Carlo (HMC) (Chen et al., 2014), and Bayes by Backpropagation (BBB) (Graves, 2011) in Appendix E.

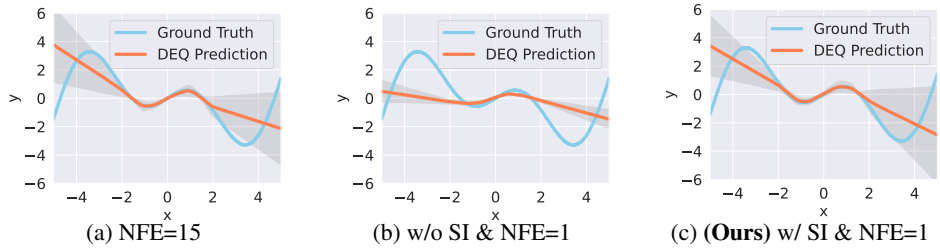
This is the first work to study uncertainty estimation for DEQ and we incorporate the point estimation SGD training for comparison, which is denoted as **SGD**. Besides, we conduct experiments with Linearized Laplacian Approximation (LLA), and Last-Layer Laplacian Approximation (LLLA) (Daxberger et al., 2021) in computer vision tasks for efficiency comparison.

Measurement. We use mean square error (MSE) to evaluate the performance of the steady-state PDE solving task, and the error rate (ERR) to evaluate classification performance. To measure the uncertainty, we adopt the negative log-likelihood (NLL) as the evaluation, which assumes a Gaussian predictive distribution and estimates the variance from multiple stochastic predictions. For the NLL computation in SGD, we equip the SGD predictor with a Gaussian observation model following (Bayer et al., 2013) (details in Appendix B). For classification tasks, we additionally compute the expected calibration error (ECE) to evaluate calibration, which measures the alignment between predictive confidence and empirical accuracy. We evaluate the efficiency of DEQs through numbers of function evaluations (NFE), where a smaller NFE indicates better efficiency. We also report the running time on hardware to reflect the practical efficiency of our method on a single A6000 GPU. Finally, we provide out-of-distribution results in Appendix F.

5.2 SYNTHETIC DATASET

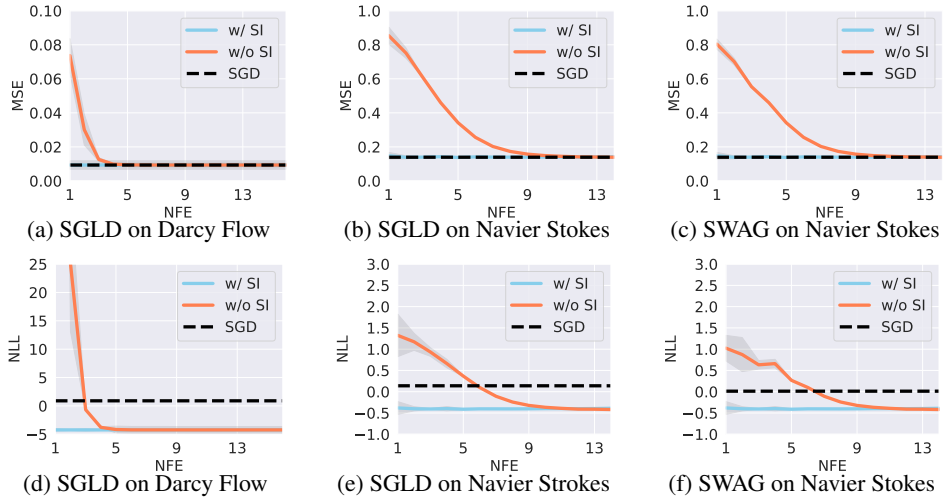
In this section, we present results on a toy dataset to visualize the uncertainty estimation using the standard deviation. The dataset is generated using the function $y = x\cos(x) + \epsilon$, where $\epsilon \sim N(0, 0.01)$. We use data points from the interval $[-2, 2]$ for training and evaluate on $[-5, 5]$. We apply MCMC to train FC-DEQ and conduct inference under three settings: (1) standard inference

[1]We abuse the notation of SGLD throughout experiments, as the preconditioned variant merely applies a scaling to the injected noise, which does not affect our method in any substantive way.



(a) NFE=15 (b) w/o SI & NFE=1 (c) **(Ours)** w/ SI & NFE=1

Figure 4: Uncertainty estimation with FC-DEQ on the toy dataset. **SI** represents our sequential inference. The gray region correspond to the predicted mean along with three standard deviations.



(a) SGLD on Darcy Flow (b) SGLD on Navier Stokes (c) SWAG on Navier Stokes

(d) SGLD on Darcy Flow (e) SGLD on Navier Strokes (f) SWAG on Navier Stokes

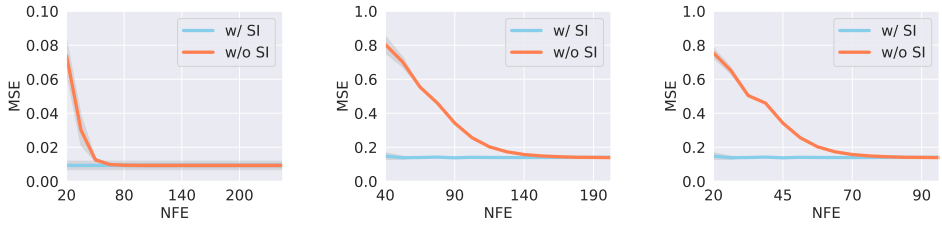
Figure 5: The Mean Squared Error (MSE) and negative log-likelihood (NLL)of FNO-DEQ on Darcy Flow and Navier Stokes. **SI** represents sequential inference.

with 15 fixed-point iterations, (2) standard inference with one fixed-point iteration, and (3) sequential inference with one fixed-point iteration. The results, shown in Figure 4, demonstrate that our method achieves accurate predictions and reliable uncertainty estimation with just one iteration, matching the performance of standard inference with 15 fixed-point iterations.

5.3 Numerical Experiments

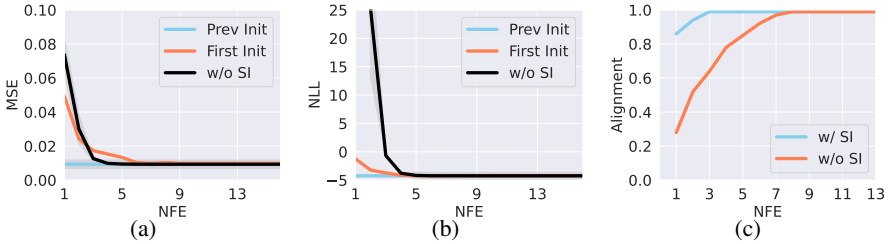
We compare the metrics between the standard inference and our sequential inference on Darcy-Flow and Navier-Stroke equations. To illustrate the effectiveness, we report the measurements for each NFE up to the maximum number of iterations. We use "**w/ SI**" to denote our sequential inference and "**w/o SI**" for standard inference. We present the results on Darcy-Flow using MCMC and SWAG with FNO-DEQ in Figures 5. Additionally, we present the running time of the experiments using SGLD in Figure 6. To compare performance across different NFEs, we present the results as curves, with the gray shaded regions indicating the mean $\pm$ one standard deviation over 5 runs. For the numerical results, please refer to Appendix G. Due to the page limitation, more results across all architectures and ImageNet are provided in Appendix D. We can draw the following observations.

Our proposed sequential inference significantly accelerates uncertainty estimation for Bayesian DEQs. The error rate and negative log-likelihood decrease more rapidly with our sequential inference on both Darcy Flow and Navier Stokes. Sequential inference consistently outperforms standard inference with the same number of iterations. For example, in Figure 5b, sequential inference requires only 1 step to achieve the same error (0.009) as the maximum number of iterations for all methods, whereas standard inference requires at least 4 steps to reach it. On Navier Stokes, sequential inference needs only 1 step (0.139) to match the performance of standard inference at 9 steps. From a runtime perspective, our method achieves substantial speedups. For FNO-DEQ-SGLD on Navier Stokes, sequential inference requires 40 seconds to reach a low error, compared to 142 seconds with standard inference, resulting in a $3\times$ acceleration. Our method not only accelerates inference but also maintains strong performance across evaluation metrics. All the Bayesian methods provide stronger performance compared to point estimation, addressing the unreliable problem.



(a) FNO-DEQ-SGLD (Darcy) (b) FNO-DEQ-SGLD (NS) (c) FNO-DEQ-SWAG (NS)

Figure 6: The Mean Squared Error (MSE) with respect to the running time of FNO-DEQ.



(a) (b) (c)

Figure 7: Ablation studies for SI. Comparison of the (a) Mean Squared Error (MSE) and (b) negative log-likelihood (NLL) between the previous fixed point and the first fixed point. (c) The ratios of the predictions are consistent with the ones with the max number of evaluations.

5.4 Ablation Study

To better understand how our sequential inference works, we conduct comprehensive ablation studies by evaluating the effectiveness of our sequential sampling and analyzing the instance-level performance. Additionally, we investigate the impact of the warm-up strategy in Appendix I and show ablation studies on the hyperparameters of the uncertainty estimation methods and Langevin Sampling in Appendix K and L. Finally, we provide memory usage study in Appendix M.

Effectiveness of Sequential Sampling. To show the effectiveness of sequential sampling, we also validate whether the previous fixed point is a better initialization compared to other options. We conduct experiments using the first fixed point in the sequence as the initialization for comparison with respect to the measurements. The results for FNO-DEQ-SGLD are shown in Figure 7. The previous fixed point consistently outperforms the first one as an initialization, showing the effectiveness of sequential sampling. In addition, we present an ablation study on the Langevin sampling step size τ in SWAG, detailed in Appendix L.

Instance-Level Test. In the main experiments, we evaluate the performance of our method on both prediction and uncertainty estimation. However, global performance alone is insufficient to fully demonstrate the effectiveness of our approach. Our method modifies the initialization, which could potentially alter the final predictions. To show that our method can closely align with standard DEQ inference, we perform an instance-level ablation study. We measure the percentage of labels that match those obtained by standard DEQ inference that have reached the fixed point. As illustrated in Figure 7c, our method quickly aligns with the standard DEQ inference, achieving an alignment rate close to 1. In contrast, without sequential inference, DEQs with the same NFE exhibit slower alignment of the final fixed points, requiring more than 7 steps.

6 Conclusion

This paper presents the first study on uncertainty estimation for DEQs, demonstrating that uncertainty estimation can pose a significant challenge for these models. To address this, we propose Bayesian DEQs and identify that the associated computational cost is substantial. We further introduce sequential sampling to capture similarities in uncertainty estimation and sequential computation to accelerate the process by mitigating the redundancy during the inference. Comprehensive experiments show that our method can achieve up to a $3\times$ speedup without any performance degradation on Bayesian DEQs. One limitation of our approach is that certain uncertainty estimation methods, such as deep ensembles, are difficult to directly apply our sequential sampling to capture the similarities. Nevertheless, our findings provide valuable insights and represent a significant step toward the trustworthy deployment of DEQs, advancing the security of real-world AI applications.

7 AI USE STATEMENT

In this work, we used generative AI tools to improve the clarity and readability of the manuscript and to identify potential logical inconsistencies in our writing and arguments. The tools were not used to generate research hypotheses, theoretical claims, proofs, methodology, experimental results, or interpretations of results. All AI-assisted suggestions were critically reviewed by the authors, and any revisions were independently assessed for correctness and consistency with the research. We take full responsibility for the final content of this work.

8 ETHICS STATEMENTS

Our paper studies Bayesian Deep Equilibrium Models and proposes sequential inference to accelerate inference speed. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted as an ethics problem here, to the best of our knowledge.

REFERENCES

Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information fusion*, 76:243–297, 2021.

Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and J Zico Kolter. Differentiable convex optimization layers. *Advances in neural information processing systems*, 32, 2019.

Brandon Amos and J Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, pp. 136–145. PMLR, 2017.

Donald G Anderson. Iterative procedures for nonlinear integral equations. *Journal of the ACM (JACM)*, 12(4):547–560, 1965.

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. Deep equilibrium models. *Advances in neural information processing systems*, 32, 2019.

Shaojie Bai, Vladlen Koltun, and J Zico Kolter. Multiscale deep equilibrium models. *Advances in neural information processing systems*, 33:5238–5250, 2020.

Shaojie Bai, Vladlen Koltun, and J Zico Kolter. Neural deep equilibrium solvers. In *International Conference on Learning Representations*, 2021a.

Shaojie Bai, Vladlen Koltun, and J Zico Kolter. Stabilizing equilibrium models by jacobian regularization. *arXiv preprint arXiv:2106.14342*, 2021b.

Shaojie Bai, Zhengyang Geng, Yash Savani, and J Zico Kolter. Deep equilibrium optical flow estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 620–630, 2022.

Rémi Bardenet, Arnaud Doucet, and Chris Holmes. On markov chain monte carlo methods for tall data. *Journal of Machine Learning Research*, 18(47):1–43, 2017.

Justin Bayer, Christian Osendorfer, Sebastian Urban, and Patrick van der Smagt. Training neural networks with implicit variance. In *International Conference on Neural Information Processing*, pp. 132–139. Springer, 2013.

Rohitash Chandra, Konark Jain, Ratneel V Deo, and Sally Cripps. Langevin-gradient parallel tempering for bayesian neural learning. *Neurocomputing*, 359:315–326, 2019.

Qi Chen, Yifei Wang, Yisen Wang, Jiansheng Yang, and Zhouchen Lin. Optimization-induced graph implicit nonlinear diffusion. In *International Conference on Machine Learning*, pp. 3648–3661. PMLR, 2022.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

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

Erik Daxberger, Agustinus Kristiadi, Alexander Immer, Runa Eschenhagen, Matthias Bauer, and Philipp Hennig. Laplace redux-effortless bayesian deep learning. *Advances in neural information processing systems*, 34:20089–20103, 2021.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

Laurent El Ghaoui, Fangda Gu, Bertrand Travacca, Armin Askari, and Alicia Tsai. Implicit deep learning. *SIAM Journal on Mathematics of Data Science*, 3(3):930–958, 2021.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, et al. A survey of uncertainty in deep neural networks. *Artificial Intelligence Review*, 56(Suppl 1):1513–1589, 2023.

Zhengyang Geng, Xin-Yu Zhang, Shaojie Bai, Yisen Wang, and Zhouchen Lin. On training implicit models. *Advances in Neural Information Processing Systems*, 34:24247–24260, 2021.

Alex Graves. Practical variational inference for neural networks. *Advances in neural information processing systems*, 24, 2011.

Fangda Gu, Heng Chang, Wenwu Zhu, Somayeh Sojoudi, and Laurent El Ghaoui. Implicit graph neural networks. *Advances in Neural Information Processing Systems*, 33:11984–11995, 2020.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pp. 1321–1330. PMLR, 2017.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

Laurent Valentin Jospin, Hamid Laga, Farid Boussaid, Wray Buntine, and Mohammed Bennamoun. Hands-on bayesian neural networks—a tutorial for deep learning users. *IEEE Computational Intelligence Magazine*, 17(2):29–48, 2022.

Malvin H Kalos and Paula A Whitlock. *Monte carlo methods*. John Wiley & Sons, 2008.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

Jeremiah Liu, Zi Lin, Shreyas Padhy, Dustin Tran, Tania Bedrax Weiss, and Balaji Lakshminarayanan. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. *Advances in neural information processing systems*, 33:7498–7512, 2020.

Xiaorui Liu, Wei Jin, Yao Ma, Yaxin Li, Hua Liu, Yiqi Wang, Ming Yan, and Jiliang Tang. Elastic graph neural networks. In *International Conference on Machine Learning*, pp. 6837–6849. PMLR, 2021.

David JC MacKay. Bayesian interpolation. *Neural computation*, 4(3):415–447, 1992.

Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. *Advances in neural information processing systems*, 32, 2019.

Tanya Marwah, Ashwini Pokle, J Zico Kolter, Zachary Chase Lipton, Jianfeng Lu, and Andrej Risteski. Deep equilibrium based neural operators for steady-state pdes. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Ashwini Pokle, Zhengyang Geng, and J Zico Kolter. Deep equilibrium approaches to diffusion models. *Advances in Neural Information Processing Systems*, 35:37975–37990, 2022.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

Burr Settles. Active learning literature survey. 2009.

Joost Van Amersfoort, Lewis Smith, Yee Whye Teh, and Yarin Gal. Uncertainty estimation using a single deep deterministic neural network. In *International conference on machine learning*, pp. 9690–9700. PMLR, 2020.

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)*, pp. 681–688. Citeseer, 2011.

Ezra Winston and J Zico Kolter. Monotone operator equilibrium networks. *Advances in neural information processing systems*, 33:10718–10728, 2020.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. Generalized out-of-distribution detection: A survey. *International Journal of Computer Vision*, pp. 1–28, 2024.

Zonghan Yang, Tianyu Pang, and Yang Liu. A closer look at the adversarial robustness of deep equilibrium models. *Advances in Neural Information Processing Systems*, 35:10448–10461, 2022.

A Proof for Theorem 1

Theorem 2 (Restated, 1) *Consider a sequence of parameter samples $\{\theta_1, \theta_2, \ldots, \theta_N\}$ and a DEQ $f(\boldsymbol{z}, \boldsymbol{x}; \theta)$. Suppose the following conditions are satisfied:*

- *f has non-sharp fixed points and is smooth within a local region $C \subset \mathbb{R}^d$, where $\{\theta_1, \theta_2, \ldots, \theta_N\} \subset C$. Specifically, there exist constants $M_1 > 0$ and $M_2 > 0$ such that for all $\theta \in C$,*

$$\left\| \left(I - \frac{\partial f}{\partial z}(\cdot; \theta) \right)^{-1} \right\|_2 \leq M_1 \quad \text{and} \quad \left\| \frac{\partial f}{\partial \theta}(\cdot; \theta) \right\|_2 \leq M_2, \tag{8}$$

- *The parameters in the sequence are sufficiently close to each other: there exists $\delta > 0$ such that for any $0 \leq i \leq N-1$,*

$$\|\theta_{i+1} - \theta_i\|_2 \leq \delta. \tag{9}$$

For any $0 \leq i \leq N-1$, the following inequality holds:

$$\|\boldsymbol{z}^*_{i+1} - \boldsymbol{z}^*_i\|_2 \leq \sqrt{d}\delta M_1 M_2,$$

where $\boldsymbol{z}^* \in \mathbb{R}^d$ is the solution of fixed-point problems.

Proof: According to the implicit function theorem, the equation $z = f(z, x; \theta)$ has an implicit function denoted by $\varepsilon(x; \theta)$. Define $\Delta\theta_i := \theta_{i+1} - \theta_i$, then we have:

$$\begin{aligned}\|z^*_{i+1} - z^*_i\|_2^2 &= \|\varepsilon(x; \theta_{i+1}) - \varepsilon(x; \theta_i)\|_2^2 \\ &= \|\varepsilon(x; \theta_i + \Delta\theta_i) - \varepsilon(x; \theta_i)\|_2^2.\end{aligned}$$

By the Mean Value Theorem, there exists a series of θ_i^j such that:

$$\begin{aligned}\|\varepsilon(x; \theta_i + \Delta\theta_i) - \varepsilon(x; \theta_i)\|_2^2 &= \sum_{j=1}^{d} \|\varepsilon_j(x; \theta_i + \Delta\theta_i) - \varepsilon_j(x; \theta_i)\|_2^2 \\ &= \sum_{j=1}^{d} \|\varepsilon_j(x; \theta_i) + \nabla_\theta \varepsilon_j(x; \theta_i^j)^T \Delta\theta_i - \varepsilon_j(x; \theta_i)\|_2^2 \\ &= \sum_{j=1}^{d} \|\nabla_\theta \varepsilon_j(x; \theta_i^j)^T \Delta\theta_i\|_2^2,\end{aligned}$$

where ε_j represents the j-th component of ε. Now, using our assumptions and applying the implicit function theorem again, we get:

$$\begin{aligned}\|z^*_{i+1} - z^*_i\|_2^2 &= \sum_{j=1}^{d} \|\nabla_\theta \varepsilon_j(x; \theta_i^j)^T \Delta\theta_i\|_2^2 \\ &\leq \sum_{j=1}^{d} \left\| \left(I - \frac{\partial f}{\partial z}(z^*, x; \theta_i^j) \right)^{-1} \right\|_2^2 \left\| \frac{\partial f}{\partial \theta}(z^*, x; \theta_i^j) \right\|_2^2 \|\Delta\theta_i\|_2^2 \\ &\leq d\delta^2 M_1^2 M_2^2,\end{aligned}$$

where z^* is the fixed point corresponding to the given inputs and parameters. □

B Experiment Details

B.1 Dataset

In this section, we introduce the dataset we use in our paper. Note that the steady-state PDE solving task is a regression task.

Steady-State PDE. We conduct experiments on two families of steady-state PDEs: the Darcy Flow equation and the steady-state Navier-Stokes equation for incompressible fluids.

(1) Darcy Flow is a form of linear elliptic PDE with in two dimensions. The PDE is defined as follows:

$$-\nabla \cdot (a(x)\nabla u(x)) = f(x), \qquad x \in (0,1)^2 \tag{10}$$

$$u(x) = 0, \qquad x \in \partial(0,1)^2. \tag{11}$$

Note that the diffusion coefficient $a(x)$ are always positive. These PDEs are used to model the steady-state pressure of fluids flowing through a porous media. They can also be used to model the stationary state of the diffusive process with $u(x)$ modeling the temperature distribution through the space with a defining the thermal conductivity of the medium. We use the dataset provided by the previous work for our experiments with steady-state Darcy-Flow Li et al. (2020). All the models are trained on 1024 data samples and tested on 500 samples. The resolution of the original images is 421×421, which we downsample to 85×85 for our experiments.

(2) Navier-Stokes equation for an incompressible viscous fluid in the vorticity form defined on a torus, i.e., with periodic boundary condition is as follows:

$$u \cdot \nabla\omega = \nu\Delta\omega + f, \qquad x \in \Omega \tag{12}$$

$$\nabla \cdot u = 0, \qquad x \in \Omega, \tag{13}$$

where $\Omega := (0, 2\pi)^2$, and u is the velocity and ν is the viscosity and f is the external force term. We train all the models on data with viscosity values $\nu = 0.01$ and $\nu = 0.001$, and create a dataset for steady-state incompressible Navier-Stokes, which we will make public as a community benchmark for steady-state PDE solvers.

Image Classification. We use MNIST, CIFAR-10, and ImageNet to evaluate the performance on the classification tasks (LeCun et al., 1998; Krizhevsky et al., 2009; Deng et al., 2009).

B.2 Model Architecture

FC-DEQ. We use fully connected deep equilibrium models (FC-DEQ) for the classification task on the toy dataset and MNIST, following (Winston & Kolter, 2020). Specifically, we define the fixed-point problem using the following formula:

$$z = \sigma(U_l x + V_l \sigma(U_{l-1} x + V_{l-1} \cdots \sigma(U_1 x + V_1 z + b_1) + b_{l-1}) + b_l), \tag{14}$$

where U_i, V_i are weight matrices, b_i are bias terms, and σ is the activation function. For our toy regression task, we set the number of layers to 2 and the dimension of the hidden embedding to 32. For MNIST, we set the number of layers to 2 and the dimension of the hidden embedding to 1000. The activation function σ of both experiments is chosen as ReLU.

FNO-DEQ. FNO-DEQ is a deep equilibrium variant of the FNO architecture that directly solves for the solution of a steady-state PDE as the infinite-depth fixed point of an implicit operator layer using a blackbox root solver and differentiates analytically through this fixed point, resulting in $O(1)$ training memory (Marwah et al., 2023). The width of an FNO layer is set to 32 across all the networks. Additionally, we retain only 12 Fourier modes in FNO layer, and truncate higher Fourier modes. We use Anderson acceleration (Anderson, 1965) as the root solver. For the backward pass, we use approximate implicit gradients (Geng et al., 2021), which are light-weight and more stable in practice, compared to implicit gradients computed by inverting Jacobian. For the backward pass, we use phantom gradients (Geng et al., 2021) which are computed as:

$$u^* = \tau G_\theta(u^*, s) + (1 - \tau)u^*, \tag{15}$$

where τ is a tunable damping factor and u^* is the fixed point computed using the Anderson solver in the forward pass. This step can be repeated S times. We use $\tau = 0.5$ and $S = 1$ for Darcy Flow, and $\tau = 0.8$ and $S = 3$ for Navier-Stokes.

MDEQ. Multi-resolution deep equilibrium models (MDEQ) are a class of implicit networks well-suited to large-scale and highly hierarchical pattern recognition tasks. They are inspired by modern computer vision deep neural networks, which leverage multi-resolution techniques to learn features at different scales. These simultaneously learned multi-resolution features allow us to train a single model on a diverse set of tasks and loss functions. For example, a single MDEQ model can be trained to perform both image classification and semantic segmentation. MDEQs have been shown to match or exceed the performance of recent competitive computer vision models, achieving high accuracy in sequence modeling and other tasks.

We report the model hyperparameters in Table 1. For CIFAR-10, we employ both MDEQ-SMALL and MDEQ-LARGE architectures, while for ImageNet, we utilize only MDEQ-SMALL. Most of the hyperparameters remain consistent with the settings specified in the work of (Bai et al., 2021b). MDEQs define several resolutions in the implicit layer, aligning with the design principles of ResNet in computer vision models. The main difference between the architectures lies in the channel size and resolution level. Additionally, all models incorporate GroupNorm, as outlined in the standard MDEQ model from (Bai et al., 2020).

B.3 Measurement Details

Mean Squared Error (MSE). MSE measures the average squared difference between the ground-truth targets and the model predictions:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 .$$

	CIFAR-10		ImageNet
	SMALL	LARGE	SMALL
Input Size	32×32	32×32	224×224
Block	BASIC	BASIC	BOTTLENECK
Number of Branches	3	4	4
Number of Channels	[8, 16, 32]	[32, 64, 128, 256]	[32, 64, 128, 256]
Number of Head Channels	[7, 14, 28]	[14, 28, 56, 112]	[28, 56, 112, 224]
Final Channel Size	200	1680	2048

Table 1: Model hyperparameters of MDEQ in our experiments.

Negative Log-Likelihood (NLL). NLL evaluates the quality of the predicted probability distribution by penalizing low probability assigned to observed outcomes:

$$\mathrm{NLL} = -\frac{1}{n}\sum_{i=1}^{n} \log p(y_i \mid x_i, \theta).$$

For the NLL of SGD point estimation, we estimate the variance with a validation dataset.

Expected Calibration Error (ECE). ECE quantifies the discrepancy between predicted confidence and empirical accuracy over M confidence bins:

$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{n} \left|\mathrm{acc}(B_m) - \mathrm{conf}(B_m)\right|,$$

where

$$\mathrm{acc}(B_m) = \frac{1}{|B_m|}\sum_{i \in B_m} \mathbf{1}(\hat{y}_i = y_i),$$

and

$$\mathrm{conf}(B_m) = \frac{1}{|B_m|}\sum_{i \in B_m} \hat{p}_i$$

denote the empirical accuracy and average confidence of bin B_m, respectively.

B.4 Training Setting

We train all the networks for 500 epochs with Adam optimizer. The learning rate is set to 0.001 for Darcy flow and 0.005 for Navier-Stokes. We use a learning rate weight decay of $1e-4$ for both Navier-Stokes and Darcy flow. The batch size is set to 32. In the case of Darcy flow, we also use cosine annealing for learning rate scheduling.

We also report the training hyperparameters for computer vision in Table 2. Following work (Bai et al., 2021b), we use Jacobian regularization to ensure stability during the training process. Moreover, to prevent overfitting, we employ data augmentation techniques such as random cropping and horizontal flipping for CIFAR-10, which are commonly utilized in various computer vision tasks.

B.5 Bayesian Inference Setting

In this section, we present the details of Bayesian inference. For MCMC, we choose Stochastic Gradient Langevin Dynamics (SGLD) as the algorithm to sample parameters from the Bayesian posterior. We strictly follow the formulation in the paper (Welling & Teh, 2011) and use the size of the training dataset to define the scaling of the noise term in Langevin dynamics (Welling & Teh, 2011). The learning rate is selected from $\{0.5, 0.1, 0.01, 0.001\}$ for both FC-DEQ and MDEQ. We set the sampling frequency to 2. For the prior distribution, we use Gaussian distribution, which is identical to weight decay. We set the weight decay as 5×10^{-4} for both MNIST and CIFAR-10. For MC Dropout, we apply variational dropout as recommended in the MDEQ training recipe (Bai et al., 2020), selecting the dropout rate from $\{0.01, 0.05, 0.1, 0.3, 0.5\}$. For SWAG, we approximate

	MNIST	CIFAR-10	
	FC-DEQ	SMALL	LARGE
Batch Size	512	96	96
Epochs	450	120	220
Learning Rate	0.01	0.001	0.001
Learning Rate Schedule	Linear	Linear	Linear
Momentum	0.0	0.98	0.98
Jacobian Reg. Strength	0	0.5	0.4
Jacobian Reg. Frequency	0	0.05	0.02

Table 2: Training hyperparameters in our experiments.

the covariance matrix with a scaling factor of 0.5, following the paper (Maddox et al., 2019). We sample 100 models for FC-DEQs and 30 models for MDEQs for both MC Dropout and SWAG.

For the implementation of the Laplace approximation, we closely follow the code outlined in Laplace Redux (Daxberger et al., 2021). We adopt both the linearized Laplace approximation and the last-layer Laplace approximation as baselines to evaluate the efficiency of uncertainty estimation.

C ALGORITHMS

In this section, we present the algorithms for MCMC and MC Dropout in Algorithm 2 and 3.

Algorithm 2 Sequential Inference for SGLD

Require: DEQ $f(\cdot)$, training epoch T, burn-in m, sampling frequency s, input data $\boldsymbol{x}$
1: Parameter samples initialization: $\Theta = [\ ]$
2: **for** $1 \leq i \leq T$ **do**
3: Train f for one epoch and get parameters θ_i
4: **if** $i \geq m$ and $mod(i, s) = 0$ **then**
5: Append θ_i to Θ
6: **end if**
7: **end for**
8: Fixed-point initialization: $\boldsymbol{z}^0 = \mathbf{0} \in \mathbb{R}^{B \times d}$
9: Prediction posterior initialization: $\mathbf{p}(\mathbf{y}|\mathbf{x}; \theta) = [\ ]$
10: **for** $1 \leq i \leq \lfloor (T - m)/s \rfloor$ **do**
11: $\boldsymbol{z}^i = \text{Solver}(f, \boldsymbol{x}, \boldsymbol{z}^{i-1}; \theta_i)$
12: Classify $\boldsymbol{z}^i$ to get $\mathbf{p}(\mathbf{y}|\mathbf{x}; \theta_\mathbf{i}) \in \mathbb{R}^C$
13: Append $\mathbf{p}(\mathbf{y}|\mathbf{x}; \theta_\mathbf{i})$ to $\mathbf{p}(\mathbf{y}|\mathbf{x}; \theta)$
14: **end for**
Return: $\mathbf{p}(\mathbf{y}|\mathbf{x}; \theta)$

Algorithm 3 Sequential Inference for MC Dropout

Require: DEQ $f(\cdot)$, dropout rate p, number of inference N, input data $\boldsymbol{x}$
1: Fixed-point initialization: $\boldsymbol{z}^0 = \mathbf{0} \in \mathbb{R}^{B \times d}$
2: Prediction posterior initialization: $\mathbf{p}(\mathbf{y}|\mathbf{x}; \theta) = [\ ]$
3: **for** $1 \leq i \leq N$ **do**
4: Apply dropout mask with rate p to f
5: $\boldsymbol{z}^i = \text{Solver}(f, \boldsymbol{x}, \boldsymbol{z}^{i-1}; \theta_i)$
6: Classify $\boldsymbol{z}^i$ to get $\mathbf{p}(\mathbf{y}|\mathbf{x}; \theta_\mathbf{i}) \in \mathbb{R}^C$
7: Append $\mathbf{p}(\mathbf{y}|\mathbf{x}; \theta_\mathbf{i})$ to $\mathbf{p}(\mathbf{y}|\mathbf{x}; \theta)$
8: **end for**
Return: $\mathbf{p}(\mathbf{y}|\mathbf{x}; \theta)$

D RESULTS ON COMPUTER VISION

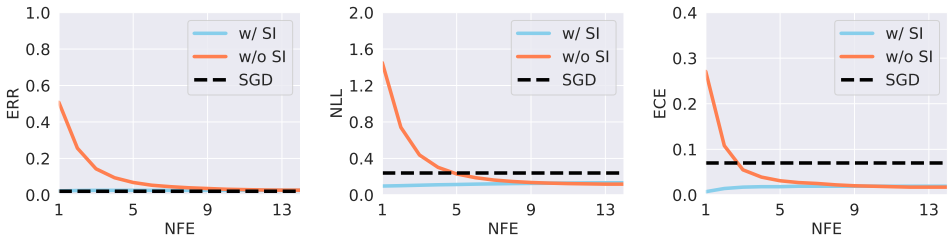


Figure 8: The error rate (ERR), negative log-likelihood (NLL), and expected calibration error (ECE) of FC-DEQ-SGLD on MNIST. **SI** represents our sequential inference.

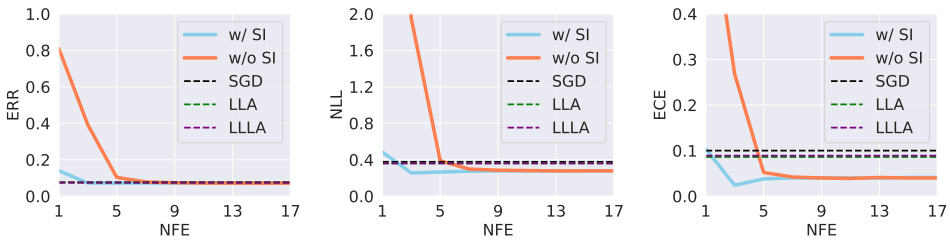


Figure 9: The error rate (ERR), negative log-likelihood (NLL), and expected calibration error (ECE) of MDEQ-LARGE-SGLD on CIFAR-10. **SI** represents sequential inference.

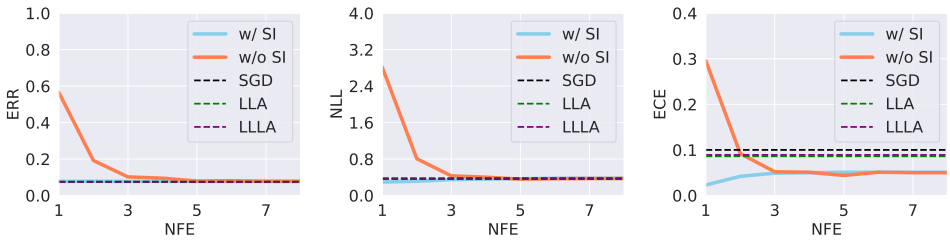


Figure 10: The error rate (ERR), negative log-likelihood (NLL), and expected calibration error (ECE) of MDEQ-LARGE-SWAG on CIFAR-10. **SI** represents sequential inference.

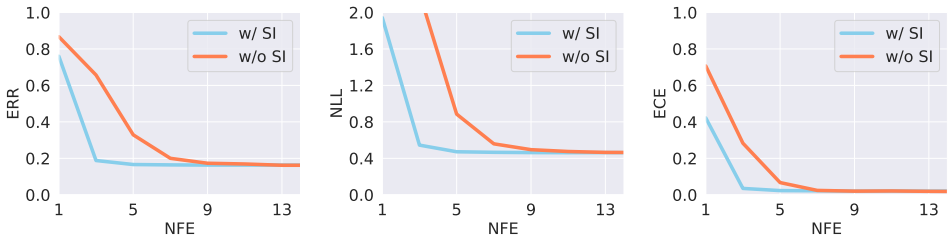


Figure 11: The error rate (ERR), negative log-likelihood (NLL), and expected calibration error (ECE) of MDEQ-SMALL-SGLD on CIFAR-10. **SI** represents our sequential inference.

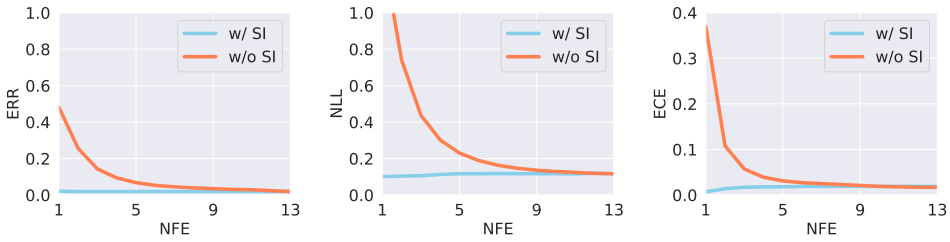


Figure 12: The error rate (ERR), negative log-likelihood (NLL), and expected calibration error (ECE) of FC-DEQ-SWAG on MNIST. **SI** represents our sequential inference.

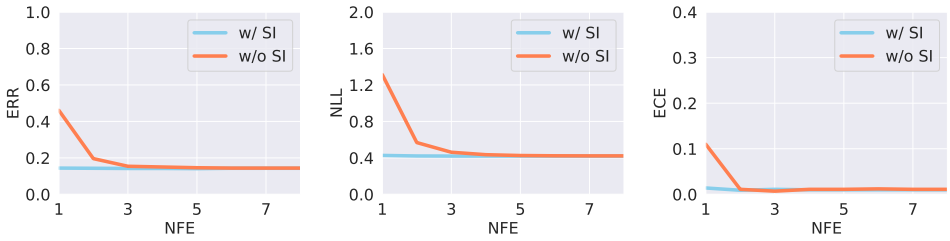


Figure 13: The error rate (ERR), negative log-likelihood (NLL), and expected calibration error (ECE) of MDEQ-SMALL-SWAG on CIFAR-10. **SI** represents our sequential inference.

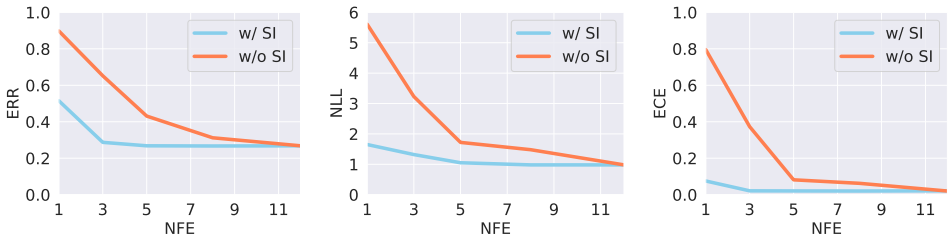


Figure 14: The error rate (ERR), negative log-likelihood (NLL), and expected calibration error (ECE) of MDEQ-SMALL-SWAG on ImageNet. **SI** represents our sequential inference.

In this section, we present additional results for computer vision tasks, illustrated in Figures 8, 9, 10 11, 12, and 13. The results demonstrate that our method can significantly accelerate uncertainty estimation across different DEQs. Specifically, the error rate decreases rapidly with our sequential inference method, requiring only a single step to achieve strong performance across all metrics. In contrast, other methods take considerably longer: MDEQ-SMALL-SGLD requires at least 9 steps, FC-DEQ-SWAG requires 9 steps, and MDEQ-SMALL-SWAG requires 4 steps. This consistent trend highlights the generalization capability of our approach.

Figure 14 presents the results on ImageNet using MDEQ-SMALL, which align with those observed on smaller datasets.

Besides, the results indicate that Laplacian Approximation offers no significant improvement over standard SGD. The suboptimal results may stem from the unrolled computation scheme and the instability of the Jacobian matrices, both of which can hinder effective approximation (Bai et al., 2021b). Moreover, the uncertainty estimation is slightly improved when NFE is small in Figure 8. This trend is similar to the findings of (Guo et al., 2017), where the negative log-likelihood increases as the training epoch increases. We argue that this behavior is a result of the accuracy-efficiency trade-off inherent in DEQs, which avoids overconfidence.

E RESULTS ON OTHER BASELINES

In this section, we present the performance of our sequential inference on other baselines.

Monte Carlo Dropout. Previous studies have shown that modern dropout techniques in deep neural networks can be interpreted as a Gaussian process, providing a theoretical basis for uncertainty estimation (Gal & Ghahramani, 2016). The training is identical to SGD with dropout, but dropout is applied during the inference phase as well. By ensembling predictions across different dropout masks, we can approximate the posterior distribution of the model.

It is not straightforward to perform sequential sampling with MC Dropout, as the algorithm already defines its sampling procedure. However, the typically low dropout rate leads to similar outputs. We approximately treat these outputs as our sequential samples. The results in Appendix D demonstrate that the standard MC Dropout setting is sufficient to capture the redundancy in the outputs.

We report results for MC Dropout in Figures 15, 16, and 17. These results show that our method can also accelerate MC Dropout, even though it does not inherently involve sequential sampling, as discussed in the main paper. For example, our sequential inference method reaches strong performance in just 1 step, whereas MC Dropout requires at least 5 steps to achieve comparable results.

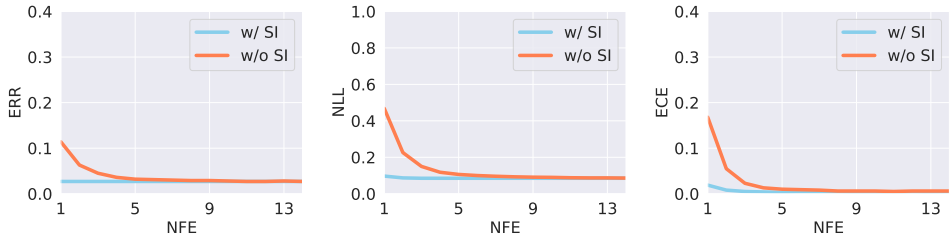


Figure 15: The error rate (ERR), negative log-likelihood (NLL), and expected calibration error (ECE) of FC-DEQ-DROP on CIFAR-10. **SI** represents our sequential inference.

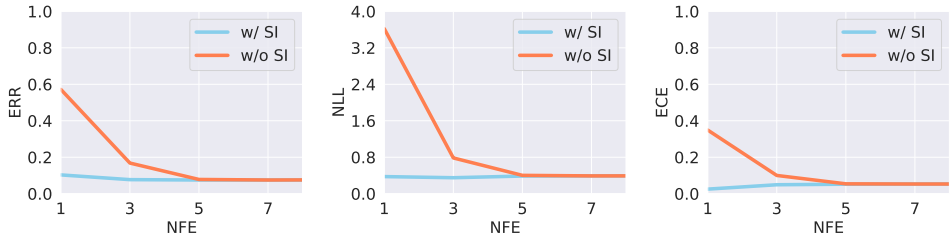


Figure 16: The error rate (ERR), negative log-likelihood (NLL), and expected calibration error (ECE) of MDEQ-LARGE-DROP on CIFAR-10. **SI** represents our sequential inference.

Deep Ensembles. Our method also works for deep ensembles. Since deep ensembles train several models with different initializations, sequential sampling is not applicable to them. We directly apply sequential computation to those pre-trained weights. The results are shown in Table 3. While its acceleration performance is not as strong as that of other Bayesian methods (as discussed in the limitations), it still achieves faster inference than the standard approach.

Hamiltonian Monte Carlo. In addition to SGLD, we conduct experiments on HMC to show that our method works for a faster convergence algorithm, HMC. As shown in Table 4, our method can consistently accelerate the convergence, and the benefit of our method is not limited to SGLD.

Bayes by Backpropagation. To show that our sequential inference works on variational inference methods, we conduct experiments with BBB. As shown in Table 5, our method can consistently improve the convergence speed based on BBB.

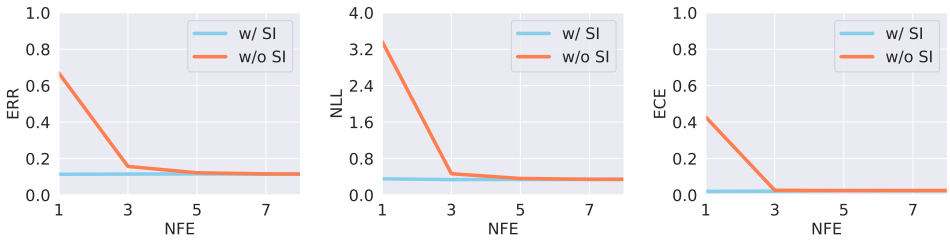


Figure 17: The error rate (ERR), negative log-likelihood (NLL), and expected calibration error (ECE) of MDEQ-SMALL-DROP on CIFAR-10. **SI** represents our sequential inference.

NFE	1	3	5	8	12
Error (Deep Ensemble)	28.2%	14.3%	4.6%	1.8%	1.8%
Error (Deep Ensemble+SI)	5.4%	1.9%	1.8%	1.8%	1.8%
ECE (Deep Ensemble)	18.3%	6.5%	2.0%	1.5%	1.5%
ECE (Deep Ensemble+SI)	10.7%	2.2%	1.6%	1.5%	1.4%

Table 3: The empirical calibration error (ECE) and error rate of Deep Ensemble using sequential inference (SI) with varying numbers of function evaluations (NFE). We report the results using FC-DEQ on MNIST.

F Results on Out-of-Distribution Detection

Out-of-distribution (OOD) detection is often viewed as a measure of uncertainty estimation. A model with robust uncertainty estimation can effectively detect whether input data is out of distribution. In this study, we report the entropy values for different models, where higher entropy indicates better performance. We utilize Fashion-MNIST (Xiao et al., 2017) as the OOD dataset for MNIST, and CIFAR-100 (Krizhevsky et al., 2009) as the OOD dataset for CIFAR-10. As shown in Table 6, our method (w/ SI) consistently outperforms the standard inference approach. For each comparison, we set the number of inferences to the maximum value.

G Numerical Results

In the main paper, we only provide the visualization of the measurements. We provide the detailed numbers in Tables 7, 8, and 9.

H The Effectiveness of Sequential Sampling

In the main paper, we use sequential sampling to sample the parameters in SWAG. To show that we do not change the representivity of the samples, we compute the Wasserstein-2 distance between the samples (the number of samples is 30), which is only 0.0013. It means that the samples from our sequential sampling can be treated as the standard sampling considering the representative property.

I The Effectiveness Warm-Up Strategy

In the main results, we employ a warm-up strategy to provide a good starting point for sequential inference. Here, we evaluate the impact of the warm-up strategy. We conduct experiments on CIFAR-10 using MDEQ-LARGE-SGLD and compare the performance of three configurations: with warm-up, without warm-up, and without sequential inference. As shown in Figure 18, the warm-up strategy significantly enhances performance compared to the setting without warm-up, particularly at the initial stages. Moreover, we observe that even without warm-up, our sequential inference consistently outperforms the standard inference.

NFE	1	3	5	8	12
Error (HMC)	31.2%	12.3%	5.8%	3.1%	2.2%
Error (HMC+SI)	4.4%	2.2%	2.2%	2.2%	2.2%
ECE (HMC)	19.3%	4.5%	2.6%	1.5%	1.5%
ECE (HMC+SI)	2.7%	1.5%	1.5%	1.5%	1.5%

Table 4: The empirical calibration error (ECE) and error rate of HMC using sequential inference (SI) with varying numbers of function evaluations (NFE). We report the results using FC-DEQ on MNIST.

NFE	1	3	5	8	12
Error (BBB)	50.6%	14.3%	6.8%	4.5%	2.9%
Error (BBB+SI)	2.4%	2.2%	2.3%	2.3%	2.4%
ECE (BBB)	27.1%	5.5%	3.1%	2.2%	1.9%
ECE (BBB+SI)	1.1%	1.8%	1.9%	1.9%	1.8%

Table 5: The empirical calibration error (ECE) and error rate of BBB using sequential inference (SI) with varying numbers of function evaluations (NFE). We report the results using FC-DEQ on MNIST.

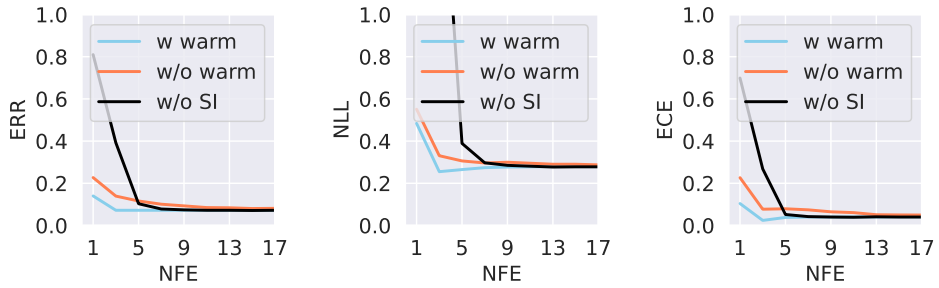


Figure 18: The error rate (ERR), negative log-likelihood (NLL), and expected calibration error (ECE) of MDEQ-LARGE-SGLD on CIFAR-10 for warm-up ablation study. "w/o SI" represents the standard DEQ inference with the same number of evaluations.

J Empirical Studies on the Assumption of the Theory

In Theorem 1, the first assumption is about the property of DEQs. To verify our assumption empirically, we estimate the M_1 and M_2 for MDEQ-SMALL trained by SGLD on CIFAR-10 (for the memory limitation, we only use the small one for illustration). We compute the maximum norm in the assumption over 1000 test samples to estimate M_1 and M_2. The estimated M_1 is 1 while M_2 is 1.6×10^{-3}. To examine a larger-scale scenario, we evaluate MCMC-SMALL trained with SWAG on ImageNet. Using 100 test samples, we obtain $M_1 = 1.002$ and $M_2 = 4.1 \times 10^{-3}$. These empirical findings suggest that the assumption made in Theorem 1 is reasonable in practice.

Models	Inference Mode	Entropy
FC-DEQ-SGLD	w/ SI	0.194
FC-DEQ-SGLD	w/o SI	0.174
MDEQ-LARGE-SGLD	w/ SI	0.402
MDEQ-LARGE-SGLD	w/o SI	0.395
MDEQ-SMALL-SGLD	w/ SI	0.346
MDEQ-SMALL-SGLD	w/o SI	0.280

Table 6: Entropy for Out-of-Distribution Data with SGLD

Table 7: Numerical Results for SGLD on Darcy Flow.

NFE	w/o SI MSE	w/o SI NLL	w/ SI MSE	w/ SI NLL
1	0.074 ± 0.010	270.837 ± 10.012	0.009 ± 0.003	-4.233 ± 0.201
2	0.030 ± 0.009	25.080 ± 12.043	0.009 ± 0.003	-4.233 ± 0.117
3	0.013 ± 0.001	-0.672 ± 0.101	0.009 ± 0.002	-4.233 ± 0.209
4	0.010 ± 0.001	-3.781 ± 0.143	0.009 ± 0.002	-4.233 ± 0.115
5	0.009 ± 0.001	-4.167 ± 0.623	0.009 ± 0.001	-4.233 ± 0.161
6	0.009 ± 0.001	-4.227 ± 0.669	0.009 ± 0.001	-4.233 ± 0.141
7	0.009 ± 0.001	-4.231 ± 0.619	0.009 ± 0.001	-4.233 ± 0.094
8	0.009 ± 0.001	-4.232 ± 0.618	0.009 ± 0.000	-4.233 ± 0.087
9	0.009 ± 0.001	-4.233 ± 0.542	0.009 ± 0.003	-4.233 ± 0.110
10	0.009 ± 0.001	-4.234 ± 0.712	0.009 ± 0.001	-4.233 ± 0.104
11	0.009 ± 0.001	-4.234 ± 0.620	0.009 ± 0.002	-4.233 ± 0.199
12	0.009 ± 0.001	-4.234 ± 0.611	0.009 ± 0.001	-4.233 ± 0.078
13	0.009 ± 0.001	-4.233 ± 0.421	0.009 ± 0.000	-4.233 ± 0.100
14	0.009 ± 0.000	-4.233 ± 0.498	0.009 ± 0.001	-4.233 ± 0.088
15	0.009 ± 0.001	-4.233 ± 0.627	0.009 ± 0.001	-4.233 ± 0.126
16	0.009 ± 0.000	-4.233 ± 0.616	0.009 ± 0.000	-4.233 ± 0.165

Table 8: Numerical Results for SGLD on Navier Stokes Equation.

NFE	w/o SI MSE	w/o SI NLL	w/ SI MSE	w/ SI NLL
1	0.854 ± 0.052	1.325 ± 0.479	0.148 ± 0.020	-0.381 ± 0.133
2	0.751 ± 0.029	1.173 ± 0.220	0.139 ± 0.009	-0.401 ± 0.051
3	0.604 ± 0.001	0.932 ± 0.115	0.140 ± 0.001	-0.406 ± 0.023
4	0.460 ± 0.003	0.661 ± 0.101	0.142 ± 0.001	-0.398 ± 0.045
5	0.342 ± 0.021	0.367 ± 0.011	0.138 ± 0.003	-0.411 ± 0.023
6	0.256 ± 0.015	0.099 ± 0.008	0.141 ± 0.002	-0.403 ± 0.014
7	0.203 ± 0.004	-0.108 ± 0.009	0.140 ± 0.001	-0.403 ± 0.018
8	0.173 ± 0.004	-0.242 ± 0.011	0.140 ± 0.001	-0.404 ± 0.015
9	0.157 ± 0.002	-0.322 ± 0.013	0.140 ± 0.000	-0.402 ± 0.011
10	0.149 ± 0.003	-0.366 ± 0.012	0.140 ± 0.001	-0.398 ± 0.010
11	0.145 ± 0.001	-0.391 ± 0.007	0.140 ± 0.001	-0.397 ± 0.012
12	0.141 ± 0.000	-0.409 ± 0.010	0.141 ± 0.000	-0.399 ± 0.011
13	0.140 ± 0.002	-0.411 ± 0.011	0.141 ± 0.001	-0.396 ± 0.007
14	0.139 ± 0.001	-0.419 ± 0.006	0.142 ± 0.001	-0.389 ± 0.009

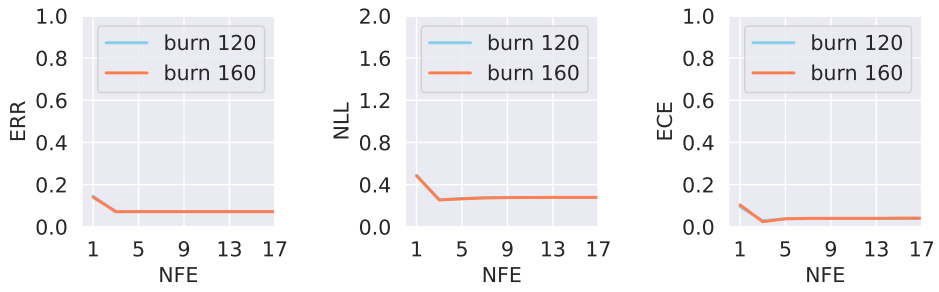


Figure 19: The error rate (ERR), negative log-likelihood (NLL), and expected calibration error (ECE) of MDEQ-LARGE-SGLD on CIFAR-10 for the ablation study of burn-in. "w/o SI" represents the standard DEQ inference with the same number of evaluations.

Table 9: Numerical Results for SWAG on Navier Stokes Equation.

NFE	w/o SI MSE	w/o SI NLL	w/ SI MSE	w/ SI NLL
1	0.804 ± 0.031	1.025 ± 0.320	0.148 ± 0.020	-0.400 ± 0.132
2	0.701 ± 0.020	0.873 ± 0.401	0.139 ± 0.009	-0.411 ± 0.049
3	0.554 ± 0.005	0.632 ± 0.108	0.140 ± 0.001	-0.431 ± 0.024
4	0.429 ± 0.001	0.571 ± 0.121	0.139 ± 0.001	-0.433 ± 0.054
5	0.301 ± 0.001	0.267 ± 0.014	0.138 ± 0.000	-0.445 ± 0.012
6	0.204 ± 0.001	0.079 ± 0.015	0.135 ± 0.003	-0.441 ± 0.011
7	0.188 ± 0.001	-0.142 ± 0.009	0.136 ± 0.002	-0.440 ± 0.009
8	0.152 ± 0.001	-0.287 ± 0.010	0.133 ± 0.001	-0.438 ± 0.012
9	0.147 ± 0.001	-0.355 ± 0.011	0.132 ± 0.000	-0.439 ± 0.011
10	0.140 ± 0.001	-0.391 ± 0.014	0.133 ± 0.001	-0.433 ± 0.010
11	0.134 ± 0.001	-0.409 ± 0.008	0.133 ± 0.001	-0.431 ± 0.007
12	0.135 ± 0.000	-0.408 ± 0.007	0.132 ± 0.000	-0.432 ± 0.008
13	0.135 ± 0.001	-0.431 ± 0.009	0.134 ± 0.001	-0.432 ± 0.005
14	0.133 ± 0.000	-0.432 ± 0.010	0.133 ± 0.001	-0.433 ± 0.009

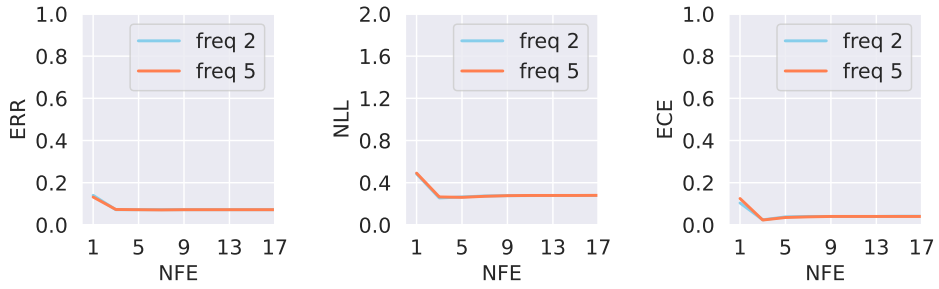


Figure 20: The error rate (ERR), negative log-likelihood (NLL), and expected calibration error (ECE) of MDEQ-LARGE-SGLD on CIFAR-10 for the ablation study of sampling frequency. "w/o SI" represents the standard DEQ inference with the same number of evaluations.

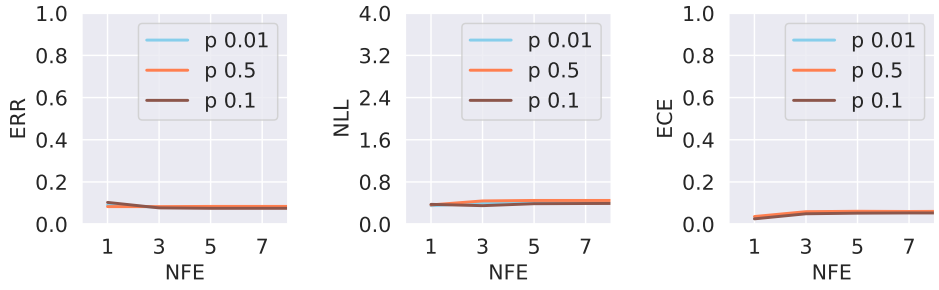


Figure 21: The error rate (ERR), negative log-likelihood (NLL), and expected calibration error (ECE) of MDEQ-LARGE-DROP on CIFAR-10 for the ablation study of dropout rate. "w/o SI" represents the standard DEQ inference with the same number of evaluations.

K Ablation of Bayesian Methods

In this section, we explore how the hyperparameters of MCMC, MC Dropout, and SWAG influence the performance of our method. As illustrated in Figures 19 and 20, we investigate how burn-in and sampling frequency impact the performance in Stochastic Gradient Langevin Dynamics (SGLD).

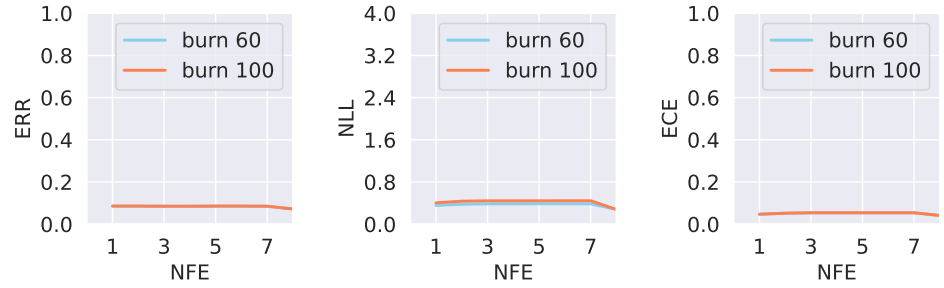


Figure 22: The error rate (ERR), negative log-likelihood (NLL), and expected calibration error (ECE) of MDEQ-LARGE-SWAG on CIFAR-10 for the ablation study of burn-in. "w/o SI" represents the standard DEQ inference with the same number of evaluations.

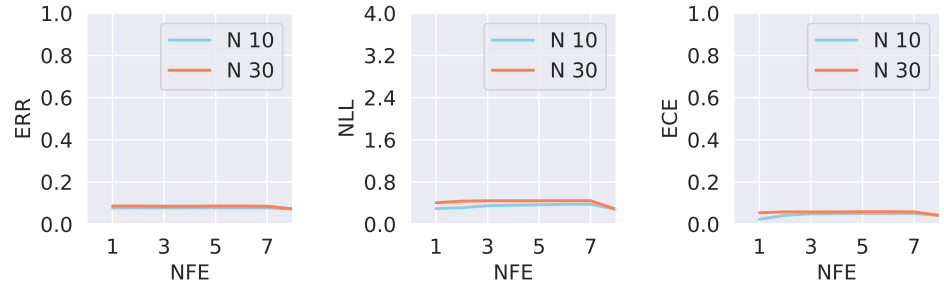


Figure 23: The error rate (ERR), negative log-likelihood (NLL), and expected calibration error (ECE) of MDEQ-LARGE-SWAG on CIFAR-10 for the ablation study of the number of sampling N. "w/o SI" represents the standard DEQ inference with the same number of evaluations.

Specifically, we set the burn-in periods to 120 and 160, and the sampling frequencies to 2 and 5. In addition, we evaluate the sensitivity of the prior in SGLD in Table 10. For MC Dropout, we analyze the effect of the dropout rate on performance, selecting dropout rates p from $\{0.01, 0.05, 0.1, 0.3\}$, with results shown in Figure 21. Additionally, we study how burn-in and the number of stored models N affect the performance in SWAG. As depicted in Figures 22 and 23, we set burn-in to 60 and 120, and N to 10 and 30. The results demonstrate that the Bayesian inference methods used are not highly sensitive to these hyperparameters within an appropriate range. Among these factors, the number of models N has the most significant impact in SWAG, where larger N has better performance, as shown in Figure 23. Across all these settings, our sequential sampling maintains similar acceleration performance.

NFE	1	3	5	8	12
prior=1	4.3%	3.5%	4.0%	3.9%	3.9%
prior=10	1.2%	1.5%	2.0%	2.0%	2.0%
prior=100	1.0%	1.4%	1.7%	1.8%	1.8%

Table 10: The empirical calibration error (ECE) of SGLD using sequential inference (SI) with varying Gaussian prior. We report the results using FC-DEQ on MNIST.

L Langevin Sampling

Metropolis–Hasting Correction. Let $\pi(x) = Z^{-1}e^{-U(x)}, \quad x \in \mathbb{R}^d$, be a probability density satisfying $\pi(x) > 0$ for all $x \in \mathbb{R}^d$, and suppose $U \in C^1(\mathbb{R}^d)$. For a fixed step size $h > 0$,

consider the Langevin proposal $Y = x - \frac{h}{2}\nabla U(x) + \sqrt{h}\,\xi, \qquad \xi \sim \mathcal{N}(0, I_d)$, whose transition density is $q_h(x,y) = (2\pi h)^{-d/2} \exp\left[-\frac{1}{2h}\left\|y - x + \frac{h}{2}\nabla U(x)\right\|^2\right]$. The unadjusted Langevin algorithm (ULA) accepts every proposal, whereas the Metropolis-adjusted Langevin algorithm (MALA) accepts the proposal with probability $\alpha(x,y) = 1 \wedge \frac{\pi(y)q_h(y,x)}{\pi(x)q_h(x,y)}$. If the proposal is rejected, the chain remains at x. Then the following statements hold.

1. Fixed-step ULA is not exact in general. The ULA transition kernel does not in general preserve π. Therefore, for fixed $h > 0$, even if the ULA chain is ergodic and converges to an invariant distribution π_h, one generally has $\pi_h \neq \pi$. For example, if $\pi = \mathcal{N}(0,1), \qquad U(x) = \frac{x^2}{2}$, then ULA satisfies $X_{k+1} = \left(1 - \frac{h}{2}\right) X_k + \sqrt{h}\,\xi_k$. For $0 < h < 4$, this chain converges to $\pi_h = \mathcal{N}\left(0, \frac{1}{1-h/4}\right)$, which differs from the desired target $\mathcal{N}(0,1)$ whenever $h > 0$. Thus a fixed-step Langevin discretization cannot in general be guaranteed to converge exactly to π.

2. The Metropolis correction makes π exactly invariant. Let P_h denote the transition kernel of MALA. For $x \neq y$, its transition density is $q_h(x,y)\alpha(x,y)$. By the definition of the Metropolis acceptance probability,

$$\begin{aligned}\pi(x)q_h(x,y)\alpha(x,y) &= \pi(x)q_h(x,y)\min\left\{1, \frac{\pi(y)q_h(y,x)}{\pi(x)q_h(x,y)}\right\} \\ &= \min\left\{\pi(x)q_h(x,y), \pi(y)q_h(y,x)\right\}.\end{aligned}$$

The right-hand side is symmetric in x and y. Hence $\pi(x)q_h(x,y)\alpha(x,y) = \pi(y)q_h(y,x)\alpha(y,x)$. Including the probability of rejection, this identity implies detailed balance, $\pi(dx)P_h(x,dy) = \pi(dy)P_h(y,dx)$. Integrating with respect to x yields $\int_{\mathbb{R}^d} \pi(dx)P_h(x,A) = \pi(A)$ for every measurable set A. Therefore $\pi P_h = \pi$. Hence the Metropolis correction removes the discretization bias at the level of the invariant distribution: the desired target π is an exact invariant distribution of MALA for every step size for which the proposal is well defined.

3. Under standard ergodicity conditions, MALA converges to π. Because the Gaussian proposal satisfies $q_h(x,y) > 0 \qquad$ for every $x, y \in \mathbb{R}^d$, and because $\pi(x) > 0$, the Metropolis acceptance probability satisfies $\alpha(x,y) > 0$. Consequently, the MALA chain can reach every set of positive Lebesgue measure with positive probability and is therefore π-irreducible. Assume in addition that the chain is aperiodic. Then, since P_h is an irreducible Markov kernel admitting the invariant probability distribution π, standard Harris-chain ergodic theory implies that P_h is positive Harris recurrent and $\|P_h^n(x,\cdot) - \pi(\cdot)\|_{\mathrm{TV}} \longrightarrow 0, \quad n \to \infty$, for π-almost every initial state x, and under the usual stronger irreducibility assumptions for all x. Equivalently, $X_n \xrightarrow{d} \pi$. Moreover, for every $f \in L^1(\pi)$, $\frac{1}{N}\sum_{k=1}^N f(X_k) \longrightarrow \int f(x)\pi(x)\,dx \quad$ almost surely. Thus MALA produces asymptotically correct Monte Carlo averages from the target distribution.

Effective Number of Langevin Sampling. To evaluate the efficiency of our Langevin Sampling, we conduct experiments to show the effective numbers. To be specific, the effective sample size quantifies the amount of independent information for correlated observations. The effective sample size can be approximated by $n_{\text{eff}} = n/[1 + 2\sum_{k=1}^{\infty}\rho_k]$, where n is the total number of observations and ρ_k is the autocorrelation at lag k. The autocorrelation is calculated as $\rho_k = \frac{\sum_{t=k+1}^n (x_t - \bar{x})(x_{t-k} - \bar{x})}{\sum_{t=1}^n (x_t - \bar{x})^2}$, where x_t denotes the observation at position (t) and $\bar{x}$ is the sample mean.

In our experiments, we evaluate the sampler on a one-dimensional Gaussian distribution using four MCMC chains, each containing 1,000 post-warm-up samples. We set $\tau = 0.01$ for Langevin sampling. The results yield an effective sample size ratio of $n_{\text{eff}}/n = 0.86$, indicating that the sampler achieves high sampling efficiency.

Step Size. For the Langevin step size τ, we conducted experiments with SWAG using MDEQ-SMALL on CIFAR-10 by evaluating the Expected Calibration Error (ECE). The results in Table 11 indicate that our method is robust to moderate variations in τ, but its performance degrades when the step size becomes excessively large since the similarities degrade.

NFE	1	3	5	8	12
$\tau = 0.001$	0.7%	1.7%	1.8%	1.8%	1.8%
$\tau = 0.005$	0.7%	1.7%	1.8%	1.8%	1.8%
$\tau = 0.01$	1.4%	1.5%	2.0%	1.9%	1.8%
$\tau = 0.05$	5.6%	2.4%	2.1%	1.8%	1.8%
$\tau = 0.1$	10.7%	3.5%	1.9%	1.8%	1.8%

Table 11: The empirical calibration error (ECE) using sequential inference (SI) with varying Langevin step size τ. We report the results using FC-DEQ on MNIST.

M MEMORY USAGE

In this section, we evaluate the running memory of our sequential inference to show that the memory overhead of sequential inference is marginal. The only additional memory usage arises from the classification process, where we retain the classification head and classification features beyond the peak memory usage of the backbone. For instance, the classification vector has only 200 dimensions, which is negligible compared to the input shape and fixed point representation in MDEQ-SMALL on CIFAR-10.

To further support this claim, we measured the memory usage of MDEQ-SMALL with a batch size of 512. The results show that the memory usage without SI is 4032 MB, while with SI it is 4244 MB, an increase of only 5.3%.