Question 1 Part 1 Report

Yan Kin Chi
yan_kin.chi@unsw.edu.au

I. Introduction

Efficient Bayesian Inference with Latent Hamiltonian Neural Networks in No-U-Turn Sampling is a paper that builds upon traditional Hamiltonian neural networks (HNNs) but introduces the following novel contributions:

1. Introducing latent variable layers within traditional HNNs to enhance the neural network's expressivity and reduce integration errors.
2. Proposed a threshold-based monitoring scheme that monitors the Hamiltonian gradients inferred by the HNN which reverts to manual integration when errors are large.

II. Paper Critique

While the paper is well-written, several crucial aspects remain unclear, which are essential for fully understanding and reproducing the presented results. To address these issues, we have organized the missing or ambiguous details into three sections corresponding to the method's three stages: generating the L-HNN training data, training the L-HNN, and performing Bayesian inference using No-U-Turn Sampling (NUTS).

A. *Generating training data for L-HNN*

- **Stochastic Coverage**: Diversity in trajectories has been emphasized to be essential to avoid training L-HNNs on limited regions of the target density. However, when sampling trajectories, q(0) is defined as the final position of the previous trajectory. This approach fails to maximize the stochastic coverage of the target density in the training data and can lead to bias towards the neighborhood of a specific density mode.
- **Notation Inconsistencies**: Defining the amount of training data to be based on "numerical gradient evaluations" without explicitly defining its meaning makes the exact training details ambiguous. This is made worse with the provided code for generating training samples only based on sample count and trajectory length.
- **Distribution Specific Parameters**: During our testing, parameters such as sample size and trajectory length can greatly alter the quality of the trained L-HNN for different distributions. However, the paper only conducts a single ablation study varying the trajectory length (exploring the critical length) for also only the 3-D degenerate Rosenbrock density. This makes understanding the specific role and significance of the parameters extremely difficult.

B. *Training the L-HNN*

- **Latent Variables λ**: The paper does not establish a clear structural relationship between the latent variables and the underlying Hamiltonian system. Furthermore, it lacks an explanation of any novel methodology for integrating latent variables into a standard multilayer perceptron framework. As a result, the proposed L-HNN effectively functions as a HNN with an output layer of λ-dimensions.

  Once again, the lack of an ablation study investigating the effect of modifying λ's dimensionality on performance limits our understanding for the significance of L-HNNs compared to traditional HNNs.

- **1-D Gaussian mixture density typo:**

  Eq. 18 (1-D Gaussian mixture density) is written as:

$$f(q) \propto 0.5\exp\left(\frac{(q-1)^2}{2\times 0.35^2}\right) + 0.5\exp\left(\frac{(q+1)^2}{2\times 0.35^2}\right)$$

  and is missing a negative sign within each exponent to match the Gaussian density form. Corrected:

$$f(q) \propto 0.5\exp\left(-\frac{(q-1)^2}{2\times 0.35^2}\right) + 0.5\exp\left(-\frac{(q+1)^2}{2\times 0.35^2}\right)$$

C. *Conducting Bayesian Inference using NUTS*

- **Excessive use of deterministic parameters**: While the inclusion of algorithm 4 and 5 provides a clear structural representation of the recursive build tree process for NUTS, the use of too many deterministic parameters makes it challenging to evaluate the generalizability of the approach. For example, thresholds Δ_{max}^{hnn} and Δ_{max}^{lf} being set to 10 and 1000 definitely may trigger unnecessary fallbacks (threshold too strict) or fail to detect errors in dynamic regions (too lenient).

III. Code Implementation & Analysis

In this section we present investigations results using a code implementation that utilises TensorFlow.

A. *Methodology and Steps to Ensure Correctness:*

To minimize implementation errors given the system's complexity, we closely replicated the original code's structure and logic using the TensorFlow framework. While alternative approaches such as leveraging TensorFlow Probability's HMC method exist and warrant future exploration, in this case we focus on maintaining fidelity to the original design.

B. *Sampling Experiments*

The method's sampling capabilities are mainly demonstrated by sampling the 1-D Gaussian mixture distribution. To obtain the training data we generated 100 trajectory samples each with $T = 50$ and $\Delta t = 0.05$. To ensure the sample count sufficiently covers all regions of the 1-D Gaussian density, we plotted the position and momentum values for all training samples as shown from in Figure 1.

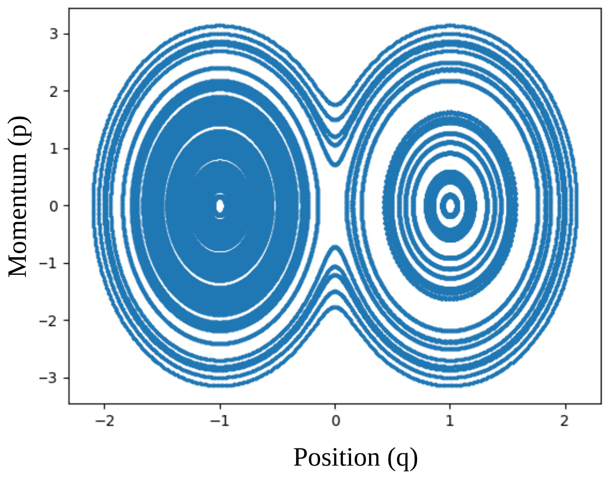

Figure 1: Phase Plot for all training samples

It can be seen that all regions of the density have been adequately covered. For training of the L-HNN we set the hidden dimension to 100, number of hidden layers to be 5, batch size of 1000, a learning rate of $5E$-4, and using a sine activation function.

Next, we generated 8000 samples from the posterior distribution using traditional HMC and L-HNN respectively and plotted the histograms for comparison.

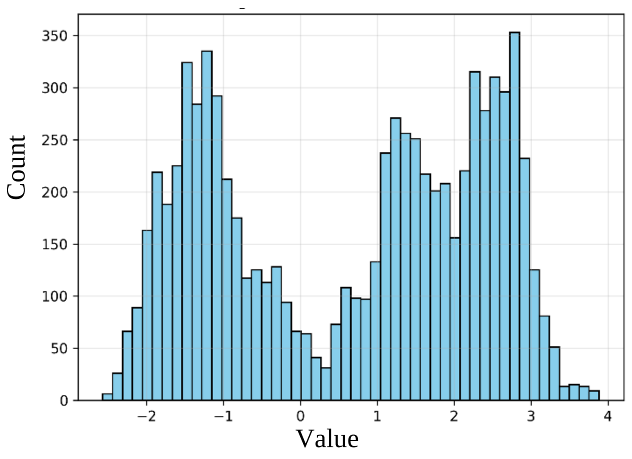

Figure 2: Sampled Distribution (L-HNN)

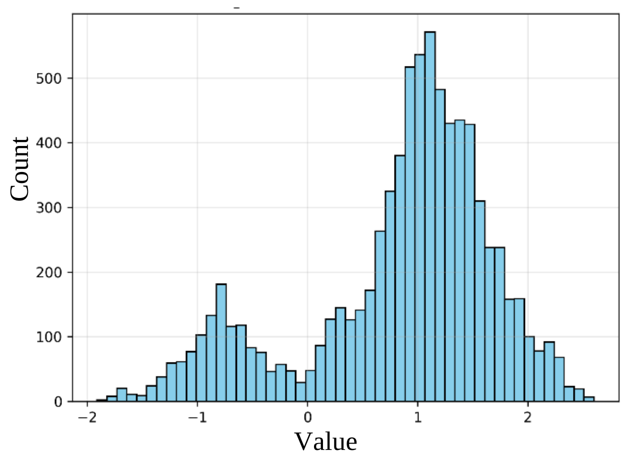

Figure 3: Sampled Distribution (HMC)

For this experiment section we were unable to fully replicate the results shown in Figures 2a, 2b, and 2c of the original paper which show the histograms of HMC and L-HNN closely matching. Specifically, there are a few notable observations:

- **Mode Sample Imbalance**: During both the generation of training samples for L-HNN (Figure 1) and the execution of NUTS, one mode consistently dominates the sample count. This is evident in Figure 1, where the left mode has disproportionately more samples, and in Figures 2 and 3, where the right mode exhibits a similar imbalance. This behaviour is expected because NUTS generates new samples by inheriting the previous position (q) while drawing a new momentum (p) from a standard Gaussian distribution. Transitioning between modes often requires traversing regions of low posterior density, effectively "climbing" over energy barriers, which makes such transitions less likely.

  To address this issue and improve exploration, multiple chains can be initiated from each available mode. This approach enhances sampling diversity and ensures better representation of the posterior distribution.

- **Compounding Trajectory Errors:** From our testing, another prominent observation is L-HNNs and by extension HNNs do not perform well when simulating long trajectories.

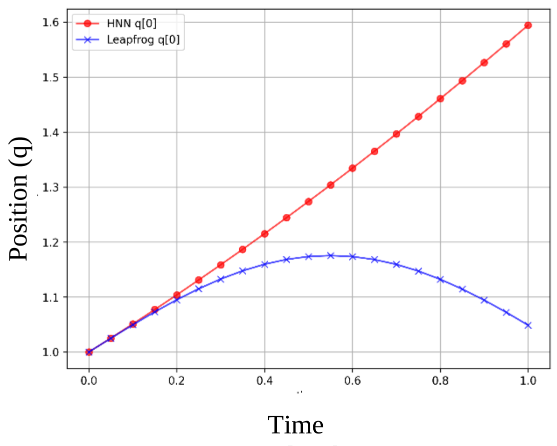

Figure 4: Simulated Trajectory

We demonstrate this by using the same trained model for the previous experiments and start at a random point in phase space. Then we employ a symplectic integrator (i.e. leapfrog) to simulate the trajectory over time using entirely either L-HNNs or HMC. As shown in Figure 4 in some cases as time passes the simulated trajectory for L-HNN (Red) can greatly deviate from the simulated trajectory for HMC (Blue). Beyond human error, this suggests that small errors in $\partial H/\partial q$ compound over time.

Overall, this reflects that neural network approaches in Bayesian inference are better suited for tasks requiring a larger number of samples with shorter trajectories. Additionally, the online error monitoring procedure is critical and could be significantly improved by making the threshold more adaptive to the target distribution. Potential improvements setting the threshold dynamically using a moving average or exponential smoothing of Hamiltonian errors from previous samples.

C. *Heavy Tail Sampling*

Within the paper, another main experimental focus is evaluating the proposed method's ability to sample from heavy-tailed distributions. Following the methodology outlined by the authors, we analyze the empirical cumulative distribution functions (eCDFs) of the generated samples. Specifically, we replicate their setup by fixing a random seed and sampling a 2-D Rosenbrock density using both L-HNN and HMC. We present the empirical cumulative distribution functions (eCDFs) for each input dimension to assess the degree of overlap and evaluate the effectiveness of the L-HNN in simulating the dynamics modeled by HMC.

For generating training data we sampled 40 trajectory samples each with $T = 250$ and $\Delta t = 0.05$. To train the L-HNN we largely inherit the same parameters setting the hidden dimension to 100, number of hidden layers to be 5, batch size of 1000, a learning rate of $5E$-4, and using a sine activation function. Finally, 8000 samples were obtained from the posterior distribution.

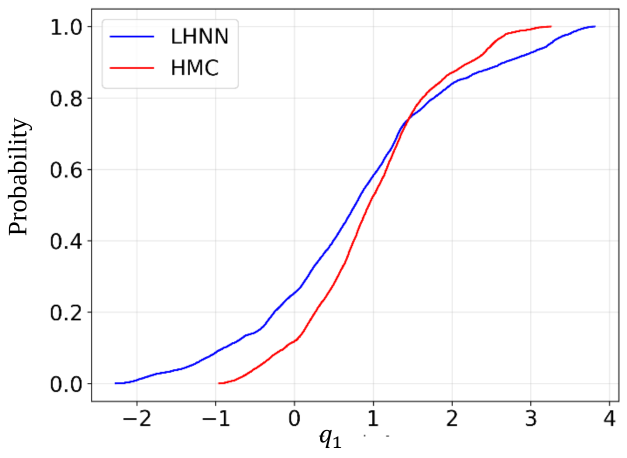

Figure 5(a): eCDF plot for Samples (1st Dimension)

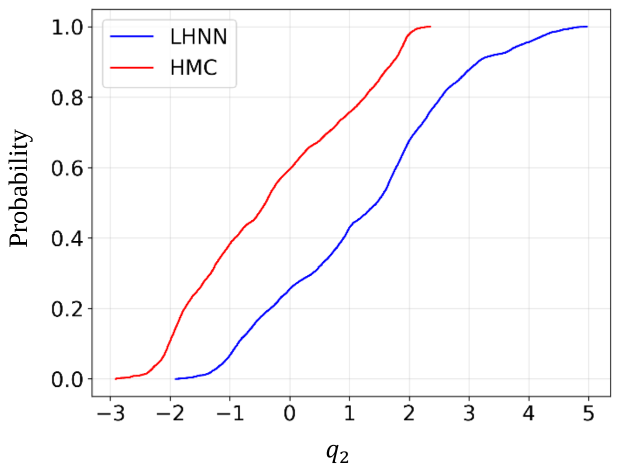

Figure 5(b): eCDF plot for Samples (2nd Dimension)

Figures 5(a) and 5(b) showcase the eCDF plots for each of the two dimensions from the 2-D Rosenbrock density. It can be seen that for dimension 1 the L-HNN (blue curve) closely follows the HMC (red curve) while diverging slightly in the tails. However for dimension 2 divergence is much more apparent especially in both tails (beyond ±3).

There is likely caused by one or both of the following reasons:

- **<u>Biased training data</u>**: As mentioned before the training data inheriting position values may cause various low-probability regions to be under-represented. For the Rosenbrock density this could be especially crucial as possess a complex "valley" like geometric landscape. This is supported from L-HNN performing well in high density regions, showcasing its ability to learn the derivative of the Hamiltonian effectively.

- **<u>Insufficient Model Capacity of MLPs:</u>** L-HNN primarily uses a multi-layer perceptron (MLP) architecture, which may lack the capacity to model the marginal and joint dependencies present in complex densities such as the Rosenbrock distribution. This is evident in Figure 5(a), which shows significantly less deviation compared to Figure 5(b), indicating a possible learning bias toward the first dimension. Alternatively, this phenomenon is caused by the second dimension representing regions with sharper curvatures, which, poses a greater challenge for the model to learn. Both interpretations point to the limited modeling capacity of MLPs.

Summarising the insights of this section, L-HNNs are an overall effective substitute to traditional HMC that can capture the primary structure of heavy tail distributions such as the N-D Rosenbrock density. Summarising the insights of this section, L-HNNs are an overall effective substitute to traditional HMC that can capture the primary structure of heavy tail distributions such as the N-D Rosenbrock density. However, for sophisticated distributions with complicated marginal and joint densities L-HNNs may struggle. Interesting areas for further research include utilising neural network architectures that are better capture cross-dimensionality dependencies such as RNNs, Transformers, or even Selective State-Space models.