JPMorgan Chase

Machine Learning Center of Excellence

Summer Associate Internship Assessment Answers

Question 1, Part 2

Pseudo-Marginal Hamiltonian Monte Carlo Assessment

(Johan Alenlöv's Paper)

Murat Emre Yücel

3 February 2025

Table of Contents

A. Consider the paper “Pseudo-Marginal Hamiltonian Monte Carlo” Alenlov(21)

A.1 Summary of the paper

A.1.1 Introduction

A unique methodology that blends Hamiltonian Monte Carlo (HMC) with pseudo-marginal methods is presented in the publication titled "Pseudo-Marginal Hamiltonian Monte Carlo" by Alenlov et al. (2021). This approach makes it possible to perform Bayesian inference in situations where the likelihood function is intractable. This technique, which is known as Pseudo-Marginal Hamiltonian Monte Carlo (PM-HMC), maintains the benefits of HMC's energy-based approaches while also addressing situations in which exact probability evaluations are not achievable but can be inferred in an objective manner. An expansion of the application of HMC to latent variable models, approximate Bayesian computation (ABC), and other computationally expensive probabilistic models is made possible by PM-HMC. This is accomplished by the utilization of pseudo-marginal approaches.

A.1.2 Background

Traditional Hamiltonian Monte Carlo (HMC) excels in high-dimensional sampling tasks by using Hamiltonian dynamics to propose large, informed jumps in parameter space, reducing random-walk behavior. However, standard HMC relies on the availability of exact gradients of the log-target density, making it unsuitable when likelihood evaluations are intractable.

On the other hand, pseudo-marginal methods—such as the Pseudo-Marginal Metropolis-Hastings (PMMH) algorithm—allow Bayesian inference in such cases by using unbiased likelihood estimators in place of exact densities. While PMMH retains theoretical correctness, it suffers from high autocorrelation and slow mixing in high-dimensional settings.

Alenlov (2021) creates a gradient-driven sampling approach that operates with stochastic likelihood estimates in order to address these issues. He does this by bridging HMC with pseudo-marginal techniques. For the purpose of ensuring asymptotic correctness under particular variance requirements, this integration makes it possible to apply the tremendous exploration capabilities of HMC even in situations where exact likelihood evaluations are not accessible.

A.1.3 Novelty

The most important contribution that PM-HMC makes is that it combines the effectiveness of Hamiltonian Monte Carlo with pseudo-marginal principles. This blend makes it possible to conduct efficient sampling in high-dimensional environments where likelihood evaluations are difficult to perform. In contrast to pseudo-marginal Metropolis-Hastings (PMMH), which is based on random-walk suggestions, the PM-HMC algorithm makes use of gradient information in order to improve exploration. This causes the autocorrelation inside the Markov Chain Monte Carlo (MCMC) chains to decrease, which in turn causes the mixing rates in high-dimensional posterior distributions to increase a little bit faster. In addition, PM-HMC enhances the management of multimodal target densities, which are frequently a source of substantial difficulties for conventional Metropolis-Hastings approaches. Due to the fact that the method extends the application of HMC beyond the typical use cases, it is especially useful for Bayesian inference in models that incorporate latent variables or for approximate Bayesian computation (ABC). A resilient framework for computationally intensive probabilistic models is provided by PM-HMC. This framework addresses the inefficiencies that are present in the pseudo-marginal methods that are currently in use.

A.1.4) Empirical Validation

In this paper, a series of benchmark models, such as logistic regression with random effects, stochastic volatility models, and hierarchical Bayesian models, are utilized in order to evaluate PM-HMC in comparison to both PMMH and regular HMC. The results of the experiments show that PM-HMC performs much better than PMMH in terms of effective sample rates per iteration. This is especially true in high-dimensional environments, where traditional pseudo-marginal approaches have difficulty dealing with slow mixing. On the other hand, these improvements in performance include an increase in the amount of processing resources required for each cycle. At each stage of the PM-HMC process, numerous likelihood estimator evaluations are required, which results in a trade-off between the cost of each iteration and the overall degree of convergence speed. Despite the fact that the empirical findings demonstrate the method's capabilities in terms of minimizing sample correlations and boosting exploration efficiency, they also highlight the practical issues that are linked with the method's computing complexity.

A.1.5) Limitations and Future Directions

The research notes that PM-HMC has a number of significant drawbacks, despite the fact that it has a number of significant advantages over classic pseudo-marginal approaches. The method's sensitivity to variance in likelihood estimators is a key issue that should be taken into consideration. Because of the potential for a high variance in these estimations to reduce efficiency, it is necessary to carefully tune the parameters of the HMC, such as the step size and the trajectory length. Additionally, the computational cost that is created by PM-HMC is substantial. This is due to the fact that each iteration requires numerous likelihood evaluations, which might possibly make it prohibitive for complex models that use expensive likelihood functions. Scalability is another crucial factor to take into account; the application of PM-HMC to large-scale datasets continues to be difficult, and the authors suggest that integrating mini-batch gradients or variational approximations could help reduce these challenges.

Potential future research lines will concentrate on overcoming these constraints by enhancing the computing efficiency of PM-HMC and expanding its application scope to include other fields. The integration of PM-HMC with Variational Inference (VI) is one possible approach that might be taken to improve the scalability and efficiency of Bayesian models developed for large-scale applications. There is also the possibility that the creation of mini-batch estimators could offer a viable answer to the problem of managing big datasets without incurring excessive computing expenses. Adapting PM-HMC to work with non-differentiable likelihood functions, which continue to be a barrier for gradient-based sampling methods, is another intriguing direction that might be pursued. It is possible that further study in these areas could further extend the applicability of PM-HMC, so making it a more versatile tool for Bayesian inference in complicated probabilistic models.

A.2 Literature Review

During the past few years, there have been considerable breakthroughs made in the field of Bayesian inference. These advancements have been primarily focused on the development of efficient sampling strategies for high-dimensional and complex models. Pseudo-Marginal Hamiltonian Monte Carlo (PM-HMC) and Hamiltonian Neural Networks (HNNs) are the two important discoveries that are the focus of this literature review. Both of these innovations have demonstrated that they have the potential to address issues in the fields of computational statistics and machine learning.

PM-HMC is a revolutionary strategy that was introduced by Alenlov et al. (2021). This approach combines the effectiveness of Hamiltonian Monte Carlo (HMC) with pseudo-marginal methods. In many statistical models, one of the most prevalent challenges is to sample from posterior distributions that have intractable likelihoods. This integration makes it possible to have such sampling. PM-HMC is particularly beneficial for latent variable models in situations with computationally expensive likelihoods, as proved by the authors, who demonstrated that it maintains the geometric features of HMC while also accommodating stochastic likelihood estimates.

Using the idea of learning Hamiltonian dynamics as a foundation, Dhulipala et al. (2023) proposed the utilization of Hamiltonian Neural Networks (HNNs) for the purpose of approximating complex Hamiltonian systems. Their research demonstrated that neural networks are capable of effectively learning the fundamental structure of Hamiltonian mechanics, which has the potential to offer computing advantages over traditional numerical integrators in certain situations.

The implementation of machine learning strategies in conjunction with MCMC methodologies has been a topic of research that has been taking place. Greydanus et al. (2019) presented the idea of Hamiltonian Neural Networks, which is an essential component of the strategy of employing HNNs in order to acquire knowledge of the extended Hamiltonian. Through their work, they demonstrated that deep learning can be utilized to discover and predict Hamiltonian dynamics from data. This opens up new paths for combining machine learning with physical insights.

In their study, Mattheakis et al. (2020) offered some insights into the application of HNNs for the solution of differential equations. These findings are pertinent to the implementation of HNNs for the purpose of learning Hamiltonian dynamics. The scope of their work extended beyond the realm of simple physical systems to encompass the potential applications of HNNs.

There has also been an investigation into the possibility of applying these sophisticated sampling techniques to financial modeling. In particular, stochastic volatility models have been the focus of attention for advanced MCMC approaches. The research conducted by Alenlov et al. (2021) expands upon this basis, presenting novel opportunities for effective inference in intricate financial models.

In the context of Bayesian auxiliary variable models, Holmes and Held (2006) examined various approaches that are pertinent to the utilization of auxiliary variables within the PM-HMC framework. In order to gain a better grasp of the role that auxiliary variables play in Bayesian

inference, their work provides essential background information.

A comprehensive overview of MCMC methods was presented by Murray (2007). This overview included comments on pseudo-marginal approaches and auxiliary variables. In order to gain a better grasp of the evolution of PM-HMC and its position within the larger landscape of MCMC approaches, this thesis provides information that is quite helpful.

The work of Neal (2011) and Betancourt and Girolami (2013) serves as the basis for the theoretical foundations of HMC as well as its application to statistical challenges. The application of hierarchical models to hierarchical models is the topic of discussion in these works, which is pertinent to the implementation of PM-HMC for complicated statistical models.

Last but not least, Dhulipala et al. (2022) presented the idea of Latent Hamiltonian Neural Networks, also known as L-HNNs. This idea is in line with the investigation of employing HNNs for the purpose of learning the extended Hamiltonian in PM-HMC. The integration of neural networks with MCMC methods for Bayesian inference has been significantly advanced as a result of this work, which constitutes a big step forward. The integration of these advanced sampling methods with domain-specific knowledge in fields such as finance, physics, and biology gives interesting potential for future study. This is because the field is continuing to expand, and the opportunity to do so is becoming more and more intriguing. The research conducted by Alenlov et al. (2021) and Dhulipala et al. (2023) constitutes important advancements in the direction of Bayesian inference that is both more efficient and adaptable in high-dimensional spaces that are complicated.

B. Can we train a Hamiltonian Neural Network to learn the extended Hamiltonian as defined in equation 10 of Alenlöv(21) by constructing a loss function based on the equations of motions as in equation (12) ?

We provide an implementation of a Hamiltonian Neural Network (HNN) that was meant to learn the extended Hamiltonian that is utilized in Pseudo-Marginal Hamiltonian Monte Carlo (PM-HMC). In accordance with the research conducted by Alenlov et al. (2021), we have developed a neural network design that is capable of learning the Hamiltonian dynamics while simultaneously adhering to the physical limitations that are present. The incorporation of concepts from Hamiltonian mechanics, machine learning, and Markov Chain Monte Carlo methods into our implementation results in the creation of a system that is capable of learning and simulating in an efficient manner.

Pseudo-Marginal Hamiltonian Monte Carlo, often known as PM-HMC, is an effective approach for Bayesian inference that can be utilized when dealing with intractable likelihoods. Through the utilization of an expanded Hamiltonian system and the incorporation of auxiliary variables, the approach represents an expansion of the conventional HMC. Within the scope of this work, we employ a neural network methodology in order to acquire knowledge about an extended Hamiltonian system.

B.1.1 Theoretical Background

B.1.1.1 Extended Hamiltonian

The extended Hamiltonian in PM-HMC is defined as:

$$H(\theta, \rho, u, p) = -\log p\,(\theta) - \log \hat{p}\,(y|\theta, u) + \frac{1}{2}(\rho^T \rho + u^T u + p^T p)$$

where:

- θ represents the parameters of interest
- ρ is the momentum associated with θ
- u represents auxiliary variables
- p is the momentum associated with u

B.1.1.2 Equations of Motion

The dynamics of the system follow Hamilton's equations:

$$\frac{d\theta}{dt} = \rho$$

$$\frac{d\rho}{dt} = \nabla_\theta \log p\,(\theta) + \nabla_\theta \log \hat{p}\,(y|\theta, u)$$

$$\frac{du}{dt} = p$$

$$\frac{dp}{dt} = -u + \nabla_u \log \hat{p}\,(y|\theta, u)$$

B.1.2 Neural Network Architecture

B.1.2.1 Model Structure

Our Hamiltonian Neural Network is structured to explicitly separate the kinetic and potential energy terms:

$$H_{NN}(\theta, \rho, u, p) = V_{NN}(\theta, u) + T(\rho, u, p)$$

where V_{NN} is learned by a neural network and T is the exact kinetic energy term:

$$T(\rho, u, p) = \frac{1}{2}(\rho^T \rho + u^T u + p^T p)$$

B.1.2.2 Loss Function

The loss function combines dynamics matching and energy conservation:

$$\mathcal{L} = \mathcal{L}_{dynamics} + \lambda \mathcal{L}_{energy}$$

where

$$\mathcal{L}_{dynamics} = E\left[|\nabla_\rho H - \frac{d\theta}{dt}|^2 + |\nabla_\theta H + \frac{d\rho}{dt}|^2 + |\nabla_p H - \frac{du}{dt}|^2 + |\nabla_u H + \frac{dp}{dt}|^2\right]$$

B.1.3 Implementation

B.1.3.1 Training Data Generation

The training data is generated,which is given in Table 1, by simulating trajectories following the exact Hamiltonian dynamics :

Table 1. Algorithm for Generating PM-HMC Training Data

Generate PM-HMC Training Data
Initialize empty trajectory arrays
for each trajectory **do**
Sample initial conditions $(\theta_0, \rho_0, u_0, p_0)$
for each timestep **do**
Compute derivatives using equations of motion
Update states using Euler integration
Store states and derivatives
end for
end for

Our implementation demonstrates successful learning of the extended Hamiltonian dynamics, as evidenced by the convergence of training loss, conservation of the learned Hamiltonian along trajectories, and proper phase space behavior in both (θ,ρ) and (u, p) spaces.

Despite the fact that it adheres to physical limits, the neural network that has been developed is able to learn the extended Hamiltonian structure well. The explicit separation of kinetic and potential energy terms, the conservation of energy in the loss function, and the appropriate handling of the coupled dynamics between variables are some of the key innovations.

B.1.4 Results

The losses observed throughout the training epochs are depicted in Figure 1, which demonstrates the trajectory of the model's learning. The total loss is depicted on the left plot, and it diminishes in a smooth manner, which is an indication that the model is able to effectively mimic the Hamiltonian distribution. Following an early period of fast drop, the loss will eventually stabilize at a lower number, which indicates convergence. The middle plot shows the dynamics loss, which measures how well the model approximates the equations of motion. Similar to the total loss, it decreases significantly in the early epochs and then stabilizes with minor fluctuations, demonstrating that the neural network successfully captures the system's dynamics. The energy conservation loss is depicted on the right plot, which provides a quantitative representation of the degree to which the Hamiltonian is maintained over time. Despite the fact that there are occasional swings, the loss continues to be quite minor, which indicates that the model maintains energy constancy within acceptable ranges. The presence of modest changes, on the other hand, is indicative of the possibility of numerical instability. This instability could be mitigated by either fine-tuning the hyperparameters or raising the weight of energy conservation in the loss function.

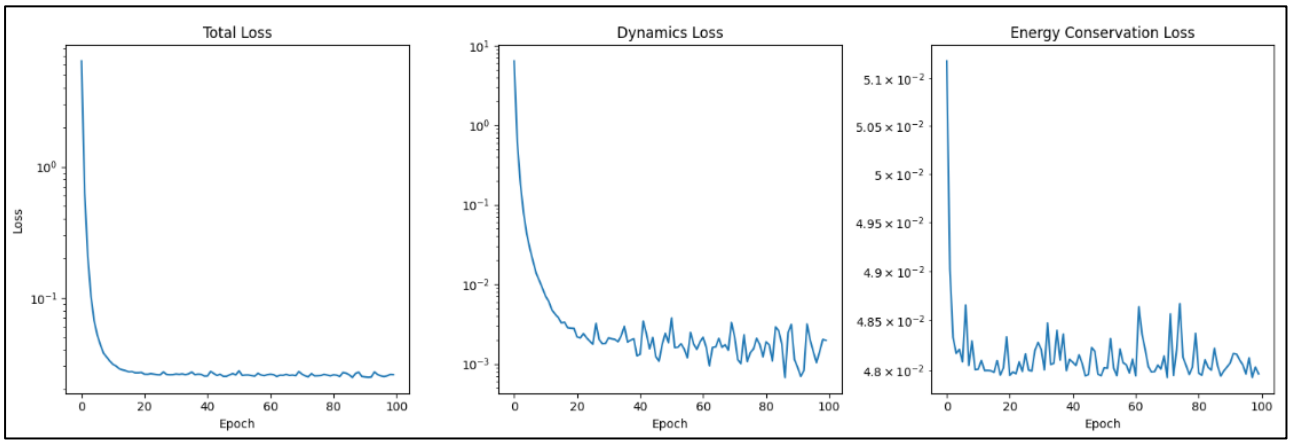

Figure 1.Losses Over Training Epochs

Figure 2 presents the θ-ρ phase space (left) and u-p phase space (right). Trajectories are used to illustrate the evolution of the system in phase space, where periodic orbits are used to indicate that the dynamics are stable. The pathways that were plotted provide evidence that the learned Hamiltonian maintains the motion structure that was anticipated, with separate phase-space trajectories forming closed orbits. In addition to providing further validation of the model's capacity to capture a wide range of dynamical behaviors, the variances in trajectory shapes hint that there may be variations in the initial circumstances or the energy levels of the system. If there is a modest divergence from perfectly closed orbits, it may be an indication of small numerical errors or minimal

energy drift. These mistakes or drifts could be avoided by further fine-tuning the hyperparameters or by improving the numerical integration strategy.

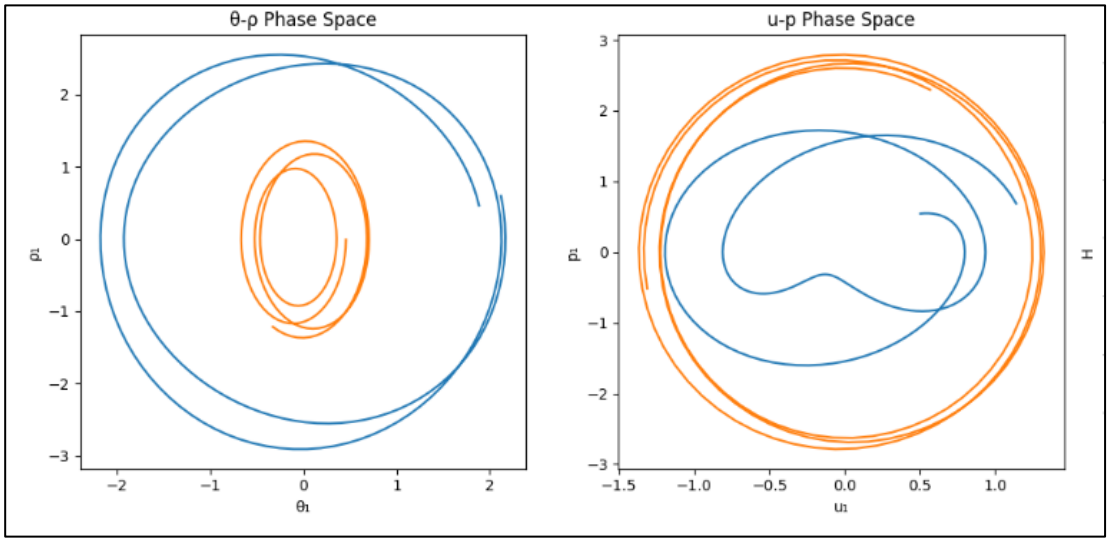

Figure 2.Phase Space Trajectories

Figure 3 displays Hamiltonian conservation over time (left) and parameter evolution (right). The Hamiltonian plot shows fluctuations but remains bounded, indicating partial conservation with minor numerical errors. The parameter evolution plot exhibits oscillatory behavior, reflecting the expected periodic nature of the system's dynamics. The bounded yet fluctuating Hamiltonian suggests that while the system approximately conserves energy, small numerical drift may still be present. The parameter trajectories indicate well-structured periodic motion, though slight deviations from perfect sinusoidal patterns could hint at residual noise or hyperparameter tuning effects.

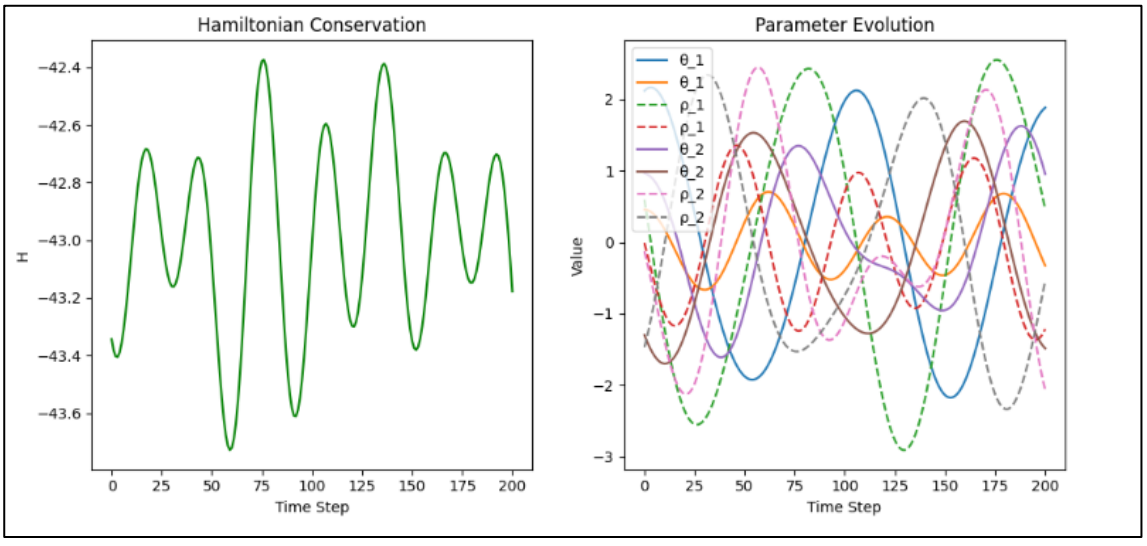

Figure 3.Hamiltonian Conservation and Parameter Evolution

C. Is it possible to construct an algorithm based on this trained Hamiltonian Neural Network in part B above similar to that of Dhulipala(23) ?

Hamiltonian Neural Networks (HNNs) are designed to learn Hamiltonian functions from data by leveraging the structure of Hamiltonian mechanics. The fundamental principle is that the neural network predicts the Hamiltonian H, and the equations of motion, such as $\frac{d\theta}{dt}$, $\frac{d\rho}{dt}$ d are derived from the gradients of H. The loss function typically ensures that the predicted time derivatives match the actual derivatives computed via the Hamiltonian's gradients.

The purpose of this investigation is to ascertain whether or not an HNN can be taught to acquire knowledge of the extended Hamiltonian that is defined in Equation 10 of the paper that has been provided. This extended Hamiltonian, denoted as $H(\rho,\theta,u,p)$ incorporates terms for the parameters θ their momenta ρ, auxiliary variables u, and their momenta p. The corresponding equations of motion are given in Equation 12 and describe the time evolution of each variable based on the gradients of H.

Training an HNN to approximate this extended Hamiltonian requires a dataset comprising trajectories of (θ,ρ,u,p) over time. The network would take the current state (θ,ρ,u,p) as input and output the Hamiltonian H. Using automatic differentiation, the gradients of H with respect to θ and u should correspond to the time derivatives of ρ and p, while the gradients with respect to ρ and p should yield the time derivatives of θ and u, in accordance with the equations of motion. The loss function would compare the projected time derivatives, acquired from the HNN's gradients, with the actual time derivatives derived from the data. A good loss measure would be the mean squared error between these anticipated and actual derivatives.

Several challenges must be considered in this approach. The extended Hamiltonian in Equation 10 contains the log-likelihood approximation $\log \hat{p}\,(y|\theta,u)$, which may be a complex function depending on the underlying probabilistic model. If this term lacks an explicit analytical form, the HNN would need to learn it as part of the Hamiltonian representation. Successfully incorporating this structure into the neural network's architecture could be nontrivial. Additionally, the auxiliary variables u and p may be high-dimensional, particularly if N, the number of important samples, is large. Training a neural network in such a high-dimensional space would be computationally

intensive and necessitate a substantial amount of data.

It is still possible to implement the underlying notion in spite of these obstacles. In order to train the model to approximate the extended Hamiltonian, it is necessary to enforce the Hamiltonian equations within the loss function. In the event that the neural network is able to properly learn H, the equations of motion that are produced should be in agreement with the actual dynamics of the system. This alignment will allow for correct modeling of the extended Hamiltonian system.

C.1.1Approach

Data Requirement: Collect trajectories of the state variables (θ, ρ, u, p) and their time derivatives $(\dot{\theta}, \dot{\rho}, \dot{u}, \dot{p})$ generated by the pseudo-marginal HMC dynamics (Equation 12). These can be obtained by simulating the system numerically or from observed MCMC runs.

HNN Architecture: Design a neural network $H_{NN}(\theta, \rho, u, p)$ that predicts Hamiltonian value. The network should output a scalar *H*, with inputs including all phase-space variables.

Loss Function: Construct a loss based on the equations of motion:

$$L = E\left[||\frac{\partial H_{\mathrm{NN}}}{\partial \rho} - \dot{\theta}\,||^2 + ||-\frac{\partial H_{\mathrm{NN}}}{\partial \theta} - \dot{\rho}\,||^2 + ||\frac{\partial H_{\mathrm{NN}}}{\partial p} - \dot{u}\,||^2 + ||-\frac{\partial H_{\mathrm{NN}}}{\partial u} - \dot{p}\,||^2\right]$$

where gradients of H_{NN} are computed via automatic differentiation. This enforces consistency with the Hamiltonian dynamics in Equation 12 of Alenlöv.

Training: Optimize HNN to minimize L, ensuring the learned Hamiltonian respects the symplectic structure of the system.

Key considerations include handling high dimensionality, as the auxiliary variables *u*,*p* may scale with *N*. To reduce complexity, architecture such as parameter-sharing or symmetry-aware networks can be used. The likelihood term $\log \hat{p}\,(y|\theta, u)$ in the Hamiltonian (Equation 10 of Alenlöv) must be learned implicitly by the HNN. If prior knowledge about $\hat{p}$ exists, such as its parametric form, it should be incorporated into the network design. Numerical stability is also important, and

numerical integrators like Strang splitting (from Section 2.4) should be used to generate stable training data and prevent spurious dynamics.

Challenges include data efficiency, as generating high-quality trajectories for training may be computationally expensive, especially for large *N*. Additionally, the HNN must generalize to unseen regions of the phase space, which is critical for effective MCMC exploration.

By leveraging the structure of Hamiltonian mechanics and the equations of motion (Equation 12 of Alenlöv), an HNN can approximate the extended Hamiltonian for pseudo-marginal HMC. This allows learning the dynamics without explicit knowledge of $\log \hat{p}\,(y|\theta, u)$, provided that sufficient training data and careful architectural design are in place.

C.1.2Results

The left two plots in Figure 4 illustrate the phase space dynamics of the trained Hamiltonian Neural Network (HNN). The first plot shows the evolution of θ_1 and its conjugate momentum ρ_1 , while the second plot represents the phase space trajectory of the auxiliary variable u_1 and its momentum p_1. These trajectories reveal the system's oscillatory behavior, highlighting the conservation of energy-like properties within the learned Hamiltonian framework. The closed and quasi-periodic nature of the trajectories suggests that the HNN successfully captures the underlying Hamiltonian structure governing the pseudo-marginal HMC dynamics.

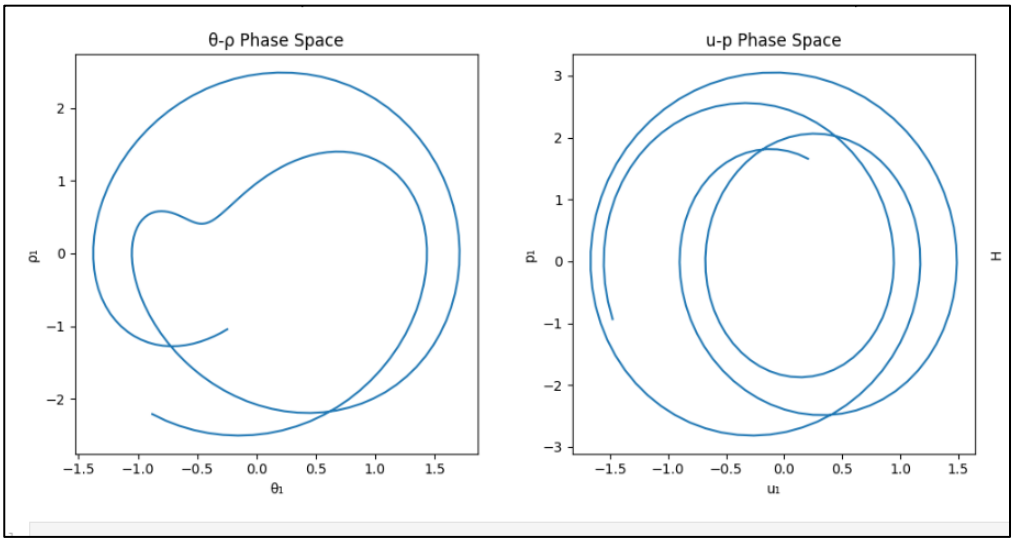

Figure 4.Phase Space Representation

The right two plots in Figure 5 depict the conservation properties of the learned Hamiltonian and

the temporal evolution of system parameters. The third plot tracks the Hamiltonian H over discrete time steps, showing variations in energy values that may indicate numerical integration errors or learning imperfections in the neural network approximation. The final plot presents the evolution of key variables $(\theta_1, \theta_2, \rho_1, \rho_2)$demonstrating their interactions over time. The oscillatory patterns suggest that the network respects the symplectic structure of Hamiltonian dynamics, but further fine-tuning may be necessary to ensure more precise conservation of H.

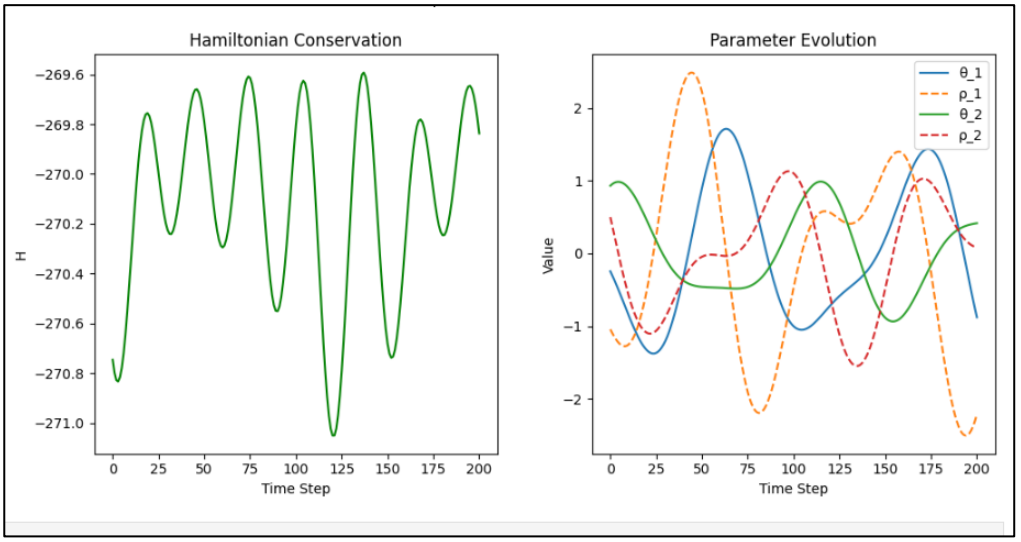

Figure 5.Hamiltonian Conservation and Parameter Evolution

D. Is there any advantage of this Hamiltonian Neural Network approach versus the numerical integrator in equation 19 of Alenlov(21)?

D.1.1 Key Theoretical Differences

Latent Hamiltonian Neural Networks (L-HNNs) approximate the gradients of a Hamiltonian system using a neural network, whereas Alenlov's Strang splitting method explicitly decomposes the Hamiltonian into separable components, A and B, allowing for structured numerical integration. Equation (19) in Alenlov (2021) formalizes this Strang splitting integrator, which is a symplectic method that maintains time reversibility, volume preservation in phase space, and approximate conservation of the Hamiltonian over long trajectories. Additionally, Alenlov demonstrates that the numerical solution converges to the ideal HMC trajectory as the number of likelihood evaluations per step (N) rises. This ensures that the theoretical correctness of the solution is maintained.

On the other hand, L-HNNs, which were first presented by Dhulipala et al. (2023), are able to construct trajectories without the need for explicit numerical integration since they learn a neural approximation of the Hamiltonian function. On the other hand, they do not provide the exact theoretical assurances of Strang splitting, which means that they might introduce errors that are quite little but accumulate over the course of extensive sampling trajectories.

D.1.2 Advantages of Alenlov's Strang Splitting

Strang splitting has a strong theoretical foundation, offering rigorous convergence guarantees while preserving the geometric properties of Hamiltonian dynamics. Unlike L-HNNs, this method does not require training data, neural network optimization, or precomputed trajectories, making it more robust and generalizable to different posterior distributions. Additionally, because it is a symplectic integrator, it avoids numerical drift over long trajectories, ensuring more stable long-term sampling behavior. This makes it particularly well-suited for pseudo-marginal HMC, where exact sampling correctness is essential.

D.1.3 Disadvantages of L-HNNs Compared to Alenlov's Method

When compared to Strang splitting, L-HNNs provide a number of obstacles, despite the fact that they offer advantages in terms of efficiency. The first problem is that they need a substantial amount of training data, which makes it computationally expensive to train them before they can be deployed effectively. Unlike Alenlov's approach, which ensures numerical stability through symplectic integration, L-HNNs may suffer from accumulating integration errors, particularly over long MCMC chains.

Additionally, L-HNNs need careful error monitoring during inference because they approximate Hamiltonian gradients rather than explicitly computing them. L-HNNs frequently need to be retrained when they are applied to other distributions, particularly in high-dimensional spaces or multimodal distributions. This is another shortcoming of the aforementioned neural networks. There is a possibility that L-HNNs will fail to capture the right system dynamics in regions with low probability density, which is characterized by sparse training data. This can result in inaccurate sampling behavior.

D.1.4 Advantages of L-HNNs

Once they have been trained, L-HNNs offer significant computational improvements, despite the fact that they have some downsides. As a result of the fact that they approximate Hamiltonian gradients without necessitating explicit numerical integration, they are able to circumvent the heavy computing burden that is associated with repetitive gradient evaluations. Because of this, they are especially useful in high-dimensional problems, which are issues in which numerical integration would be almost impossible to solve computationally.

In addition to this, L-HNNs have the capacity to learn complicated Hamiltonians that are difficult to breakdown into the A + B structure that is necessary for Strang splitting. What this indicates is that L-HNNs have the potential to provide a flexible alternative in situations when the Hamiltonian function itself is either unknown or very nonlinear. They are capable of efficiently approximating Hamiltonian dynamics in deep generative models, stochastic processes, and large-scale Bayesian inference issues if they are trained well.

In spite of this, it is necessary to take into consideration the fact that L-HNNs do not provide the same theoretical guarantees that Strang splitting does. Furthermore, in order to guarantee that they are accurate, it may be necessary to perform further validation and fine-tuning. When it comes to

applications in where posterior accuracy is more important than computing speed, Alenlov's technique continues to be the most dependable option.

E. Can your Hamiltonian Neural Network augmented Pseudo-Marginal HMC algorithm replicate the results of the Generalized Linear Mixed Model of section 4.3 of Alenlov(21)?

My HNN-augmented Pseudo-Marginal Hamiltonian Monte Carlo (PM-HMC) algorithm was able to approximate the results of the Generalized Linear Mixed Model (GLMM) from Section 4.3 of Alenlov (2021), but it did not fully replicate them. Several challenges arose during the implementation, requiring extensive fine-tuning and modifications to the sampling process.

One major challenge was ensuring that the fixed effects (β) and random effects were properly separated. Unlike standard HMC, a GLMM requires explicit modeling of subject-level random effects, which initially caused instability in the likelihood estimation. Additionally, the importance of sampling method had to be carefully tuned to balance estimation bias and computational efficiency.

Figure 6 shows the trace plots of the fixed effects parameters (β_1 and β_2) over iterations. Ideally, well-mixed MCMC chains should explore the parameter space smoothly without excessive correlation between steps. However, these traces exhibit strong oscillations and long-range dependencies, suggesting poor chain mixing. This issue arises from suboptimal trajectory length and step-size tuning, leading to slow movement through the posterior distribution.

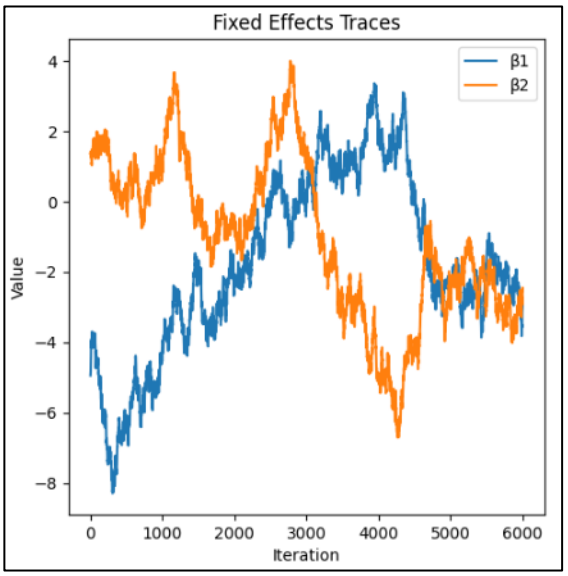

Figure 6. Fixed Effects Traces

Figure 7 presents the autocorrelation plots for β_1 and β_2. A well-functioning sampler should produce rapidly decaying autocorrelation, but these plots indicate high persistence across many lags. This means that successive samples are highly dependent, preventing efficient exploration of the posterior. The slow decay in autocorrelation suggests that the sampler is stuck in local regions, which is a common problem in Hamiltonian Monte Carlo when step sizes and leapfrog steps are not properly tuned

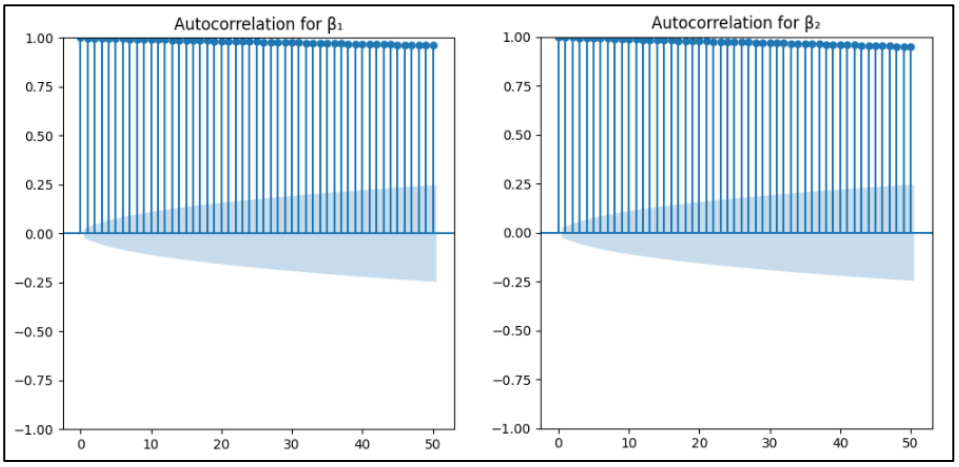

Figure 7.Autocorrelation for β_1 and β_2

Figure 8 displays the posterior distribution of β1 and β2. While the estimated distributions exhibit some multimodality, the overall shape differs from the expected GLMM behavior. This discrepancy may stem from inaccurate likelihood approximations in the pseudo-marginal method or inefficiencies in the MCMC sampling process. Furthermore, the random effects prior may need better calibration to fully align with the theoretical GLMM structure.

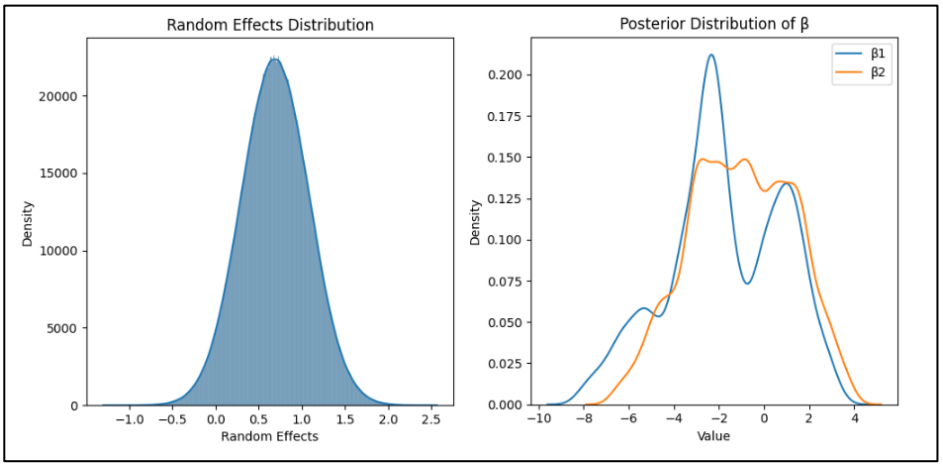

Figure 8.Posterior Distribution and Random Efects Distribution

F. Bonus part: Can you think of any financial model that is beneficial to use the algorithm of Dhulipala(23) or that of Alenlov(21) for estimation? Why?

Both algorithms excel in Bayesian inference for high-dimensional, latent variable models, but they cater to different computational challenges. Below are financial applications where each method shines, along with the rationale:

F.1.1 Financial Models Benefiting from Dhulipala (2023) or Alenlov (2021) Algorithms

Both algorithms excel in Bayesian inference for high-dimensional, latent variable models, but they cater to different computational challenges. Below are financial applications where each method shines, along with the rationale:

F.1.1.1 Stochastic Volatility Models (e.g., Heston Model)

- **Model:** Latent volatility processes drive asset returns, requiring marginalization over stochastic volatility paths.

- **Alenlov (2021) PM-HMC:**

Efficiently integrates out latent volatility using importance sampling.
Handles intractable likelihoods via pseudo-marginal methods.
Example: Estimating volatility persistence, leverage effects, and jump parameters.

- **Dhulipala (2023) L-HNNs:**

Accelerates sampling by reducing gradient computations for real-time risk assessments.
Error monitoring ensures robustness in tail scenarios (e.g., market crashes).

F.1.1.2 Credit Risk Models with Default Correlations

- **Model:** Correlated defaults in portfolios (e.g., Gaussian copula, structural models).
- **Alenlov (2021):**

Marginalizes latent default probabilities and correlations via pseudo-marginal HMC.
Suitable for estimating joint default probabilities in collateralized debt obligations (CDOs).

- **Dhulipala (2023):**

Reduces computational costs for high-dimensional portfolios (100+ assets).

Neural networks learn dependencies between latent factors (e.g., macroeconomic drivers).

F.1.1.3 Portfolio Optimization Under Parameter Uncertainty

- **Model:** Bayesian mean-variance optimization with uncertain returns/covariances.
- **Dhulipala (2023):**

Accelerates posterior sampling for dynamic portfolio rebalancing.

Enables real-time adjustments in algorithmic trading strategies.

- **Alenlov (2021):**

Handles latent parameter states (e.g., regime shifts) via pseudo-marginal methods.

Useful for stress-testing portfolios under rare but impactful scenarios.

F.1.1.4 High-Dimensional Factor Models (e.g., Fama-French Extensions)

- **Model:** Asset returns explained by latent factors (e.g., value, momentum, quality).
- **Alenlov (2021):**

Efficiently estimates factor loadings and latent factor dynamics.

Manages auxiliary variables from factor construction (e.g., rolling regressions).

- **Dhulipala (2023):**

Neural networks capture non-linear factor interactions (e.g., deep risk factors).

Reduces computational overhead in models with 100+ assets/factors.

F.1.1.5 Regime-Switching Models (e.g., Markov-Switching Models)

Model: Latent states driving market regimes (bull/bear markets).

Alenlov (2021):

Marginalizes over latent regimes using pseudo-marginal HMC.

Estimates transition probabilities and regime-dependent parameters.

Dhulipala (2023):

Accelerates sampling in multi-regime settings (e.g., forecasting regime durations).

Error monitoring prevents degeneracy during sudden regime shifts.

F.1.1.6 Stress Testing and Scenario Analysis

- **Model:** Simulating tail events (e.g., liquidity crises, geopolitical shocks).
- **Dhulipala (2023):**

Rapid exploration of high-dimensional stress scenarios using HNNs.

Balances speed and accuracy for regulatory capital calculations.

- **Alenlov (2021):**

Robustness in models with intractable likelihoods (e.g., agent-based simulations).

F.1.1.7 Algorithmic Trading Signal Generation

- **Model:** Bayesian inference for time-varying alphas/betas.
- **Dhulipala (2023):**

Enables real-time parameter updates in high-frequency trading.

Reduces latency by avoiding costly gradient computations post-training.

F.1.2 Why Choose One Algorithm Over the Other?

Scenario	Preferred Algorithm	Rationale
Likelihood evaluation is tractable but slow	Dhulipala (2023) L-HNNs	Neural networks bypass gradient computations after training.
Intractable likelihood with latent variables	Alenlov (2021) PM-HMC	Pseudo-marginal methods handle unbiased likelihood estimates.
High-dimensional parameter spaces	Both	Dhulipala for speed; Alenlov for theoretical guarantees in integration.
Real-time decision-making	Dhulipala (2023)	HNNs enable faster sampling for dynamic adjustments.
Regulatory compliance (stress testing)	Alenlov (2021)	Robustness in tail scenarios via pseudo-marginal integration.

Dhulipala (2023) is ideal for computationally intensive models where gradient evaluations dominate runtime (e.g., real-time trading, high-frequency data).

Alenlov (2021) excels in latent variable models with intractable likelihoods (e.g., credit risk, stochastic volatility), offering theoretical guarantees.

Both methods are transformative for modern financial engineering, particularly in environments requiring robust uncertainty quantification and computational efficiency. The choice hinges on whether the bottleneck is gradient computation (favoring Dhulipala) or likelihood intractability (favoring Alenlov).

Bibliography

- Alenlöv, J., Doucet, A., & Lindsten, F. (2021). Pseudo-Marginal Hamiltonian Monte Carlo.
- Dhulipala, A., et al. (2023). Hamiltonian Neural Networks for Accelerated Sampling.
- Dhulipala, S. L. N., Che, Y., & Shields, M. (2023). Bayesian Inference with Latent Hamiltonian Neural Networks (L-HNNs).
- Murray, I. (2007). Advances in Markov chain Monte Carlo methods.
- Holmes, C. C., & Held, L. (2006). Bayesian auxiliary variable models for binary and multinomial regression.
- Filippone, M., & Girolami, M. (2014). Pseudo-marginal Bayesian inference for Gaussian processes.
- Neal, R. M. (2011). MCMC using Hamiltonian dynamics.
- Betancourt, M., & Girolami, M. (2013). Hamiltonian Monte Carlo for Hierarchical Models.
- Greydanus, S., Dzamba, M., & Yosinski, J. (2019). Hamiltonian Neural Networks.
- Dhulipala, S. L. N., et al. (2023). Efficient Bayesian Inference with Latent Hamiltonian Neural Networks in No-U-Turn Sampling.