Poisson-Algebraic Parallel Scan: A Fast Symplectic Framework for Neural Hamiltonians

Jiwoong Kim JOSHUASMILE@KOREA.AC.KR
Erdembileg Davaasuren ERDEMBYLEG@KOREA.AC.KR
Youngsuk Lee YOUNGSUK@KOREA.AC.KR
Sungwoo Park SUNGWOO_PARK@KOREA.AC.KR
University of Korea, Seoul, Republic of Korea

Editor: Editor's name

Abstract

Hamiltonian neural networks (HNNs) inherit physical inductive biases but remain constrained by sequential computation which limits their scalability. We introduce an algebraic framework that embeds the Hamiltonian representation learned by the network into a set of polynomial generators within a Poisson algebra, yielding a Lie group of flows with inherently associative composition. This structural property directly enables parallel scan (prefix-sum) algorithms, thereby reducing computational complexity from $\mathcal{O}(M)$ to $\mathcal{O}(\log M)$ while preserving symplectic consistency. Empirical results confirm significant runtime and memory improvements over sequential baselines, with the scaling benefit clearly observable over thousands of steps. Our approach highlights how Hamiltonian learning can be accelerated through parallel scan, supported by a Poisson algebraic structure. Consequently, this establishes a scalable foundation for extended timescale simulation.

Keywords: Hamiltonian systems, Parallel scan, Acceleration, Lie group, Poisson algebra

1. Introduction

Hamiltonian dynamics offer a universal mathematical framework for describing the evolution of physical systems in phase space, whose symplectic structure ensures that trajectories conserve invariants such as energy and volume (Arnold, 1989). Based on this principle, HNNs were proposed as data-driven models that learn Hamiltonian function directly (Greydanus et al., 2019). Unlike black-box dynamics models, HNNs learn inductive biases aligned with the underlying physics from data, which makes them appealing for AI4Science tasks where generalization beyond training data is crucial. However, a critical bottleneck remains: HNNs rely on classical ODE-based numerical integrators that advance solutions through sequential time stepping (Chen et al., 2018). Predicting a horizon of M steps entails sequential dependence, which fixes the computational cost to grow linearly with M. As a result, in domains such as molecular dynamics (Vander Meersche, 2024) and astronomy simulations (Kim et al., 2014), the cumulative cost of such sequential scaling quickly becomes prohibitive.

In this work, we address this limitation by applying the parallel scan algorithm (Hillis and Steele, 1986; Blelloch, 1990) to Hamiltonian generation problems, thus enabling highly efficient parallel computation across horizons of M steps within a principled algebraic framework. By decomposing the Hamiltonian into polynomial generators and embedding them within a Poisson algebra, we construct a finite-dimensional Lie group of discretized flows

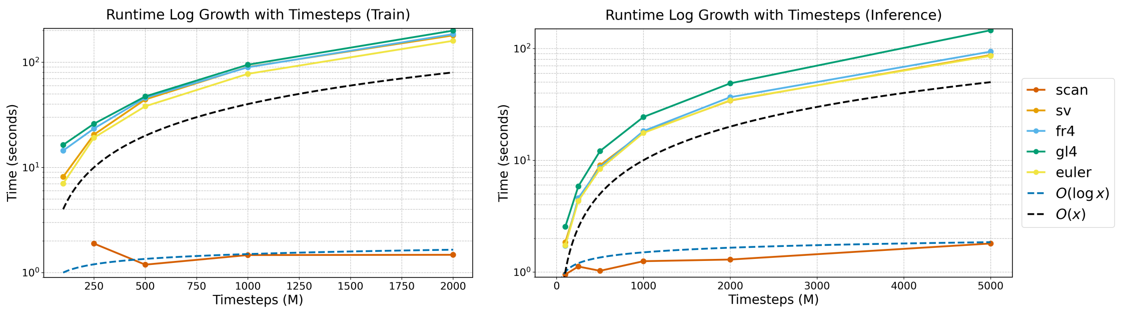

Figure 1: **Runtime Comparison.** The graph shows the runtime spent for training *(left)* and inference *(right)* time. As seen in the graph, our model shows $\mathcal{O}(log(M))$ time complexity to timestep, while others show $\mathcal{O}(M)$ time complexity.

whose composition is inherently associative. From this algebraic perspective, long trajectories can be viewed as successive compositions of short-time dynamics, which in our formulation are realized as explicitly associative operations. This associativity naturally makes it possible to apply parallel scan, reducing the computational depth to logarithmic while keeping the same number of overall calculation. Moreover, lunder the governing dynamical laws, each step remains a valid symplectomorphism (Marsden and Ratiu, 1999).

Empirical evidence shown in Figure 1 highlights the pronounced runtime gap. While classical numerical integrators exhibit $\mathcal{O}(M)$ scaling with respect to the number of timesteps, our proposed method follows $\mathcal{O}(logM)$ scaling. This divergence is already evident in few thousand timestep scales, substantiates the computational advantages of our framework.

2. Methodology

We construct Hamiltonian generators as polynomial expressions, thereby facilitating a systematic linear expansion over a finite polynomial basis that approximates the Hamiltonian function and its induced flow. To further enhance expressiveness, we organize the construction within a directed acyclic graph (DAG), which allows flexible combinations of higher-order features. Building on this foundation, we lift the polynomial generators into a Neural–Poisson Lie Group Walk. Its inherent associativity directly enables the use of parallel scan algorithms for efficient computation of Hamiltonian flows at large scale.

2.1. Polynomization of Hamiltonian Generators and Directed Acyclic Graph

We first delineate the Hamiltonian Generator hypothesis space as $\mathcal{H} \coloneqq \mathrm{Poly}_r \ni g_\theta \approx H_{\mathrm{True}}$, so that all subsequent constructions rest on a clear and consistent foundation.

Given an arbitrary phase space with coordinates $z = (q_1, \ldots, q_n, p_1, \ldots, p_n) \in \mathbb{R}^{2n}$, for the i^{th} unit vector $\mathbf{e}_i$, let the polynomial multi-indices $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{N}^n$ and coefficient neural network $\{\mathbf{c}_{\alpha,\beta} \coloneqq c_{\alpha,\beta}(z, t; \theta)\} \subset \mathcal{C}_\theta$, the Hamiltonian dynamics can be explicitly defined as

$$\dot{z} = J\nabla_z g_\theta(z) = \left[\dot{q}_i = \frac{\partial g_\theta}{\partial p_i} = \sum_{\boldsymbol{\alpha},\boldsymbol{\beta}} c_{\boldsymbol{\alpha},\boldsymbol{\beta}} \beta_i q^{\boldsymbol{\alpha}} p^{\boldsymbol{\beta}-\mathbf{e}_i},\ \dot{p}_i = -\frac{\partial g_\theta}{\partial q_i} = -\sum_{\boldsymbol{\alpha},\boldsymbol{\beta}} c_{\boldsymbol{\alpha},\boldsymbol{\beta}} \alpha_i q^{\boldsymbol{\alpha}-\mathbf{e}_i} p^{\boldsymbol{\beta}}\right]^T$$

By embedding our polynomial model space $P^{\theta}_{\le}$ into Poisson algebra, which naturally supports closed operations like pointwise product $f \star_\theta g := \pi_{\le r}(fg)$ and the canonical Poisson bracket $\{f, g\}_\theta := \pi_{\le r}\big(\{f, g\}\big)$, one can get more expressive and powerful representation of space $\big(P^{\theta}_{\le r}, \star_\theta, \{\cdot,\cdot\}_\theta\big)$ named *Neural Poisson Algebra*[1]. The closure of two operations guarantees that the result still remains in $P^{\theta}_{\le r}$ with finite application of operations.

With this algebra, we organize the procedure to create a directed acyclic graph (DAG) of Hamiltonian generators for rich representability, that collects all possible combinations of Neural Poisson Algebra while assuring the strict closure in $P^{\theta}_{\le r}$. From the base set $\mathcal{G}^{(0)} = P^{\theta}_{\le r}$, constructing each depth-k layer with applying exactly two operations $(\star_\theta, \{\cdot,\cdot\}_\theta)$ to every ordered pair in the previous layer, yielding $\mathcal{G}^{(k)}$. The procedure can be written with the formal definition

$$\mathcal{G}^{(k)} = \Big\{ \{\pi_{\le r}(f \star_\theta g)\} \bigcup \{\pi_{\le r}(\{f, g\}_\theta)\} \,/ \sim \;\Big|\; \forall f, g \in \mathcal{G}^{(k-1)} \Big\}, \quad \mathcal{G}^{(0)} = P^{\theta}_{\le r},$$

following three properties below for elimination due to algebraic equivalence and cutoff r.

$$f \star_\theta g \sim g \star_\theta f, \quad \{f, g\}_\theta \sim -\{g, f\}_\theta, \quad \{f, \{g, h\}_\theta\}_\theta + \{g, \{h, f\}_\theta\}_\theta + \{h, \{f, g\}_\theta\}_\theta \sim 0.$$

With unions of $\mathcal{G}^{(k)}$, we obtain $\mathcal{G}^{(\le K)} := \bigcup_{k=0}^{K} \mathcal{G}^{(k)}$, naming the DAG of depth K.

2.2. Neural–Poisson Lie Group Walk and Parallel Scanning

To systematically leverage our DAG-indexed structure $\mathcal{G}^{(\le K)}$ as Hamiltonian generators to approximate arbitrary Hamiltonian dynamics, we introduce the truncated Lie operator $\mathcal{L}_{g,\le r} f := \pi_{\le r}\big\{f, g\big\}$. Building on this, we can formally define the resulting family of DAG-indexed Lie transforms, a genuine Lie group , indexed by elements of the $\mathcal{G}^{(\le K)}$ as

$$\mathscr{T}_{r,K} := \Big\{ \Phi_{\epsilon,g} := \exp\big(\epsilon \mathcal{L}_{g,\le r}\big) = \sum_{j=0}^{r} \tfrac{\epsilon^j}{j!} \mathcal{L}^{j}_{g,\le r} \;\Big|\; g \in \mathcal{G}^{(\le K)} \Big\}.$$

From DAG $\mathcal{G}^{(\le K)}$, we construct an ordered sequence of Hamiltonian generators $\mathbf{g} = \big[g_1, \dots, g_m, \dots, g_M\big]$ with $|\mathcal{G}^{(\le K)}|/M$ random selections for each generator g_m. Each g_m corresponds to small-time symplectomorphism Φ_{ϵ,g_m} representing a tiny update that incrementally captures the Hamiltonian evolution over finite time horizons. We can interpret the ordered application of each small-time flow $\Phi_{\epsilon,g_m \in \mathbf{g}}$ as a discrete walk on $\mathscr{T}_{r,K}$ as $\mathcal{W}_{\mathbf{g}} : \{0, 1, \dots, M\} \to \mathscr{T}_{r,K}$ with $\mathcal{W}_{\mathbf{g}}(0) := \mathbf{Id}$, $\mathcal{W}_{\mathbf{g}}(m) := \Phi_{\epsilon,g_m} \circ \mathcal{W}_{\mathbf{g}}(m-1)$ for $1 \le m \le M$.

Let $z_0 = (q_0, p_0)$ be the initial phase–space state and fix a finite, ordered dictionary $\mathbf{g}$. The discrete time-evolution is then realized by successively applying the exact time ϵ flows generated by each g_j and its corresponding phase-space state z_j as follows:

$$\begin{tikzcd} z_0 \arrow[r, "\Phi_{\epsilon,g_1}"] \arrow[rrrrr, dashed, bend right, "\mathcal{W}_{\mathbf{g}}(M)"'] & z_1 \arrow[r, "\Phi_{\epsilon,g_2}"] & z_2 \arrow[r] & \cdots \arrow[r] & z_{M-1} \arrow[r, "\Phi_{\epsilon,g_M}"] & z_M \end{tikzcd}$$

$$\begin{aligned} z_m &= \Phi_{\epsilon,g_m} z_{m-1}, \\ z_M &= \big(\Phi_{\epsilon,g_M} \circ \cdots \circ \Phi_{\epsilon,g_1}\big) z_0, \\ z_M &= \mathcal{W}_{\mathbf{g}}(M) z_0 = (q_M, p_M). \end{aligned}$$

1. Note the projection $\pi_{\le r} \colon C^\infty(M) \to P_{\le r}$ simply discards every monomial whose total degree exceeds r.

Let $U_m := \Phi_{\epsilon, g_m}$ denote the m-th elementary Lie group update in the walk $\mathcal{W}_{\mathbf{g}}(m)$. Then, a *parallel prefix-scan* Blelloch (1990) seeks the inclusive prefixes within $O(\log M)$ parallel depth, with utilizing the assosiativity property on the operator.

$$\mathcal{W}_{\mathbf{g}}(m) := U_m \circ U_{m-1} \circ \cdots \circ U_1, \qquad m = 1, \ldots, M,$$

The Lie group composition is associative, so we can evaluate all walks simultaneously by constructing a balanced binary tree of *temporary composites* as follows:

$$V_m^{(0)} := U_m, \quad V_m^{(1)} := U_{2m} \circ U_{2m-1}, \quad V_m^{(\ell)} := V_{2m}^{(\ell-1)} \circ V_{2m-1}^{(\ell-1)}, \quad \ell = 2, \ldots, \lceil \log_2 M \rceil,$$

Recursively evaluating these composites up to $\ell = \lceil \log_2 M \rceil$ yields every prefix in parallel while preserving all symplectic invariants. With these computational acceleration tools, we now turn to the training methodology supporting proposed model. With given initial conditions $\hat{z}_0 \sim P_{\text{init}}$ and corresponding observed trajectory data $\hat{z}_t \sim P_{t>0}$, the objective is to globally infer the generator g_θ. Formally, our training objective can be expressed as:

$$\mathcal{L}(\theta) = \min_{g_\theta \in \mathcal{G}} \mathbb{E}_{\hat{z}_0, t} \left[\mathbb{E}_{\hat{z}_t} \| z_t(g_\theta) - \hat{z}_t \|^2 \middle| \hat{z}_0 \right] = \min_\theta \mathbb{E}_{\hat{z}_0, t} \left[\mathbb{E}_{\hat{z}_t} \| \mathcal{W}_{\mathbf{g}}(\lceil T/t \rceil) \hat{z}_0 - \hat{z}_t \|^2 \middle| \hat{z}_0 \right]$$

The model trains an optimal hamiltonian generator $g_\theta \in \mathcal{G}$, not on comparing hamiltonians but by an objective function that minimizes the expected squared error between predicted trajectory $z_t(g_\theta) = \mathcal{W}_g([T/t])\hat{z}_0$ and true trajectory $\hat{z}_t$.

3. Experiments

We consider the harmonic oscillator system with $H(q,p) = \frac{p^2}{2m} + \frac{1}{2} m\omega^2 q^2$, comparing with HNN model applying Euler method, Störmer–Verlet integrator (Verlet, 1967), Forest–Ruth 4th-order integrator (Forest and Ruth, 1990), and Gauss–Legendre 4th-order Runge–Kutta integrator (Sanz-Serna, 1988). Table 1 shows the overall result of our experiments.

Integrator / Method	**Train / Infer**	**GPU Mem**	**ADE / FDE** $\downarrow$	$\Delta E \downarrow$	$\Delta\omega \downarrow$
(A) Hamiltonian Neural Networks + Classical (Symplectic) Integrators					
Euler	86.0 / 159.8	12.6	$3.1{\times}10^{-3}/3.3{\times}10^{-3}$	$2.0{\times}10^{-1}$	$4.8{\times}10^{-1}$
Störmer–Verlet	87.4 / 179.8	13.6	$3.1{\times}10^{-3}/3.3{\times}10^{-3}$	$2.0{\times}10^{-3}$	$4.8{\times}10^{-2}$
Forest–Ruth 4th	93.8 / 184.8	14.2	$7.4{\times}10^{-4}/8.9{\times}10^{-4}$	$4.0{\times}10^{-4}$	$2.0{\times}10^{-2}$
Gauss–Legendre 4th	145.7 / 199.7	24.3	$6.1{\times}10^{-4}/7.1{\times}10^{-4}$	$1.3{\times}10^{-4}$	$6.3{\times}10^{-3}$
(B) Hamiltonian Neural Networks + Poisson-Algebraic Parallel Scan					
Ours	**1.5 / 1.8**	**4.7**	$\mathbf{2.9{\times}10^{-5}/3.8{\times}10^{-5}}$	$\mathbf{8.9{\times}10^{-4}}$	$\mathbf{1.3{\times}10^{-2}}$

Table 1: Quantitative Comparison of Computational Efficiency, Accuracy, and Physical Consistency

All models were trained with training step $M = 2000$ and inference step $M = 5000$. As shown in Figure 1, unlike conventional integrators whose runtime grows linearly with time steps, our framework maintains logarithmic time growth while preserving physical consistency, supported by comparable ΔE and $\Delta\omega$.

4. Conclusion

We introduced a Hamiltonian construction grounded in Lie group and Poisson algebra structures. This design endows modern neural Hamiltonian models with associative flow composition, which enables parallel scan algorithms. The result is a shift from sequential long-horizon simulation to logarithmic-depth computation. Our experiments confirm that the benefits are already evident at practical horizons, with reduced runtime and memory overhead while maintaining strict symplectic consistency. These findings show that algebraic organization of learned dynamics offers a scalable path for Hamiltonian learning. Looking forward, the framework can be extended to large-scale physical simulations, where the cost of sequential integration has long been a major bottleneck, which in turn highlights its potential as a general tool for accelerating scientific discovery.

References

Vladimir I. Arnold. *Mathematical Methods of Classical Mechanics*, volume 60 of *Graduate Texts in Mathematics*. Springer, New York, 2 edition, 1989. ISBN 978-0-387-96890-4.

Guy E. Blelloch. Prefix sums and their applications. Technical Report CMU-CS-90-190, Computer Science Department, Carnegie Mellon University, 1990.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. *arXiv preprint arXiv:1806.07366*, 2018. URL https://arxiv.org/abs/1806.07366.

E. Forest and R. D. Ruth. Fourth-order symplectic integration. *Physica D: Nonlinear Phenomena*, 1990.

Samuel Greydanus, Misko Dzamba, and David Yosinski. Hamiltonian neural networks. *NeurIPS*, 2019.

Joachim Hilgert and Karl-Hermann Neeb. *Structure and Geometry of Lie Groups*. Springer Monographs in Mathematics. Springer, New York, 1 edition, 2012. ISBN 978-0-387-84793-1. doi: 10.1007/978-0-387-84794-8. URL https://link.springer.com/book/10.1007/978-0-387-84794-8.

W. Daniel Hillis and Guy L. Steele. Data parallel algorithms. In *Proceedings of the 1st ACM Symposium on Parallel Algorithms and Architectures*, 1986.

Ji-hoon Kim, Tom Abel, Oscar Agertz, Greg L. Bryan, Daniel Ceverino, Charlotte Christensen, Charlie Conroy, Avishai Dekel, Nickolay Y. Gnedin, Nathan J. Goldbaum, Javiera Guedes, Oliver Hahn, Alexander Hobbs, Philip F. Hopkins, Cameron B. Hummels, Francesca Iannuzzi, Dusan Keres, Anatoly Klypin, Andrey V. Kravtsov, Mark R. Krumholz, Michael Kuhlen, Samuel N. Leitner, Piero Madau, Lucio Mayer, Christopher E. Moody, Kentaro Nagamine, Michael L. Norman, Jose Onorbe, Brian W. O'Shea, Annalisa Pillepich, Joel R. Primack, Thomas Quinn, Justin I. Read, Brant E. Robertson, Miguel Rocha, Douglas H. Rudd, Sijing Shen, Britton D. Smith, Alexander S. Szalay, Romain Teyssier, Robert Thompson, Keita Todoroki, Matthew J. Turk, James W. Wadsley, John H. Wise, Adi Zolotov, and AGORA Collaboration. The agora high-resolution galaxy simulations comparison project. *Astrophysical Journal Supplement Series*, 210(1):14, Jan 2014. doi: 10.1088/0067-0049/210/1/14. URL https://ui.adsabs.harvard.edu/abs/2014ApJS..210...14K/abstract.

Jerrold E. Marsden and Tudor S. Ratiu. *Introduction to Mechanics and Symmetry*, volume 17 of *Texts in Applied Mathematics*. Springer, New York, 2 edition, 1999. ISBN 978-0-387-98643-4.

J. M. Sanz-Serna. Runge-kutta schemes for hamiltonian systems. *BIT Numerical Mathematics*, 1988.

Yannick Vander Meersche. Atlas: protein flexibility description from atomistic molecular dynamics simulations. *Nucleic Acids Research*, 52(D1):D384–D392, 01 2024. doi:

10.1093/nar/gkad1084. URL https://academic.oup.com/nar/article/52/D1/D384/7438909.

Loup Verlet. Computer "experiments" on classical fluids. i. thermodynamical properties of lennard-jones molecules. *Physical Review*, 1967.

Appendix A. Mathematical Background

This appendix is for outlining the algebraic framework that underlies our approach. By adopting the Poisson algebra perspective on $(\mathbb{R}^{2n}, \omega)$, we describe dynamics in a form where Hamiltonians act as generators through the bracket structure. Projecting these generators into a finite polynomial space provides a controlled approximation that retains essential algebraic relations while remaining tractable for analysis. The induced flows continue with respecting symplectic geometry, that is, it ensures fundamental invariants are preserved despite truncation. These properties together lead to the construction of the Neural–Poisson Lie Group Walk, with algebraic closure and symplecticity for robustness and associativity for combining components to enable reliable parallel computation of local dynamics. In this way, the abstract algebraic setting connects directly to our algorithmic design, highlighting both its mathematical grounding and its role in parallel scanning.

A.1. Hamiltonian and Poisson Algebra

Poisson Bracket In Hamiltonian mechanics, the phase space observables naturally form a Poisson algebra. In this perspective, the Hamiltonian function is distinguished as the generator of dynamics. So one can interpret the Hamiltonian as a distinguished element of the Poisson algebra, and determine the temporal evolution of any observable via the bracket operation. Let $(\mathbb{R}^{2n}, \omega)$ be a $2n$-dimensional symplectic space with canonical coordinates

$$z := (\mathbf{q}, \mathbf{p}) = (q_1, \ldots, q_n,\, p_1, \ldots, p_n),$$

and standard Darboux form $\omega = \sum_{i=1}^{n} dq_i \wedge dp_i$. For a sufficiently smooth function $g \in C^\infty(\mathbb{R}^{2n})$ (a *Hamiltonian generator*), the Hamiltonian vector field X_g is defined by $\iota_{X_g}\omega = dg$, which in canonical coordinates reads

$$X_g = \sum_{i=1}^{n} \frac{\partial g}{\partial p_i}\,\partial_{q_i} - \frac{\partial g}{\partial q_i}\,\partial_{p_i}.$$

The associated *canonical Poisson bracket* on $\mathbb{R}^{2n}$ is

$$\{f, g\} := \sum_{i=1}^{n} \left(\frac{\partial f}{\partial q_i}\frac{\partial g}{\partial p_i} - \frac{\partial f}{\partial p_i}\frac{\partial g}{\partial q_i} \right) = \langle \nabla_z f,\, J\,\nabla_z g \rangle, \quad J = \begin{bmatrix} 0 & \mathbf{I}_n \\ -\mathbf{I}_n & 0 \end{bmatrix}.$$

In particular, for the coordinate map z one has $\{z, g\} = J\nabla_z g$, so the Hamiltonian flow generated by g satisfies $\dot{z} = \{z, g\}$.

Hamiltonian neural networks (HNNs) approximate an unknown scalar Hamiltonian $H_{\text{true}} : \mathbb{R}^{2n} \to \mathbb{R}$ by a neural model $g_\theta : \mathbb{R}^{2n} \to \mathbb{R}$. The learned Hamiltonian generator g_θ induces the Hamiltonian dynamics via the Poisson structure:

$$\dot{z} = \{z, g_\theta\} = J\nabla_z g_\theta(z) \;\approx\; J\nabla_z H_{\text{true}}(z) = \{z, H_{\text{true}}\}.$$

Thus, the bracket $\{\cdot, \cdot\}$ provides the map from a candidate Hamiltonian to its induced vector field, and learning g_θ means learning a generator whose Poisson-induced flow matches the observed dynamics, approximating the original Hamiltonian.

Projection and Truncation Since $g \in C^\infty(\mathbb{R}^{2n})$ is placed in the infinite dimensional space, the need for projecting and truncating the function into a finite-dimensional space arises. Let the phase space variable $z = (q_1, \ldots, q_n, p_1, \ldots, p_n) \in \mathbb{R}^{2n}$ and use multi-indices $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{N}^n$ with

$$q^{\boldsymbol{\alpha}} := \prod_{i=1}^{n} q_i^{\alpha_i}, \quad p^{\boldsymbol{\beta}} := \prod_{i=1}^{n} p_i^{\beta_i}, \quad \deg(q^{\boldsymbol{\alpha}} p^{\boldsymbol{\beta}}) := |\boldsymbol{\alpha}| + |\boldsymbol{\beta}|.$$

For a fixed cutoff $r \in \mathbb{N}$, define the finite-dimensional polynomial space

$$P_{\leq r} := \operatorname{span}\left\{q^{\boldsymbol{\alpha}} p^{\boldsymbol{\beta}} : |\boldsymbol{\alpha}| + |\boldsymbol{\beta}| \leq r\right\}.$$

The degree-r projection $\pi_{\leq r} : C^\infty(M) \to P_{\leq r}$ acts as

$$\pi_{\leq r}\left(\sum_{\boldsymbol{\alpha},\boldsymbol{\beta}} c_{\boldsymbol{\alpha},\boldsymbol{\beta}}\, q^{\boldsymbol{\alpha}} p^{\boldsymbol{\beta}}\right) := \sum_{|\boldsymbol{\alpha}|+|\boldsymbol{\beta}| \leq r} c_{\boldsymbol{\alpha},\boldsymbol{\beta}}\, q^{\boldsymbol{\alpha}} p^{\boldsymbol{\beta}}.$$

We then define truncated algebraic operations on $P_{\leq r}$ by post-composition with $\pi_{\leq r}$:

$$f \star_\theta g := \pi_{\leq r}(fg), \quad \{f, g\}_\theta := \pi_{\leq r}(\{f, g\}), \quad f, g \in P_{\leq r}.$$

By construction, $P_{\leq r}$ is closed under both $\star_\theta$ and $\{\cdot, \cdot\}_\theta$.

On this truncated space, we introduce *neural coefficients* $\{\mathbf{c}_k := c_k(z, t; \theta)\} \subset \mathcal{C}_\theta$ to parameterize a degree-r Hamiltonian:

$$P_{\leq r}^{\theta} := \operatorname{span}_{\mathcal{C}_\theta}\left\{q^{\boldsymbol{\alpha}} p^{\boldsymbol{\beta}} : |\boldsymbol{\alpha}| + |\boldsymbol{\beta}| \leq r\right\}, \quad \text{basis size } \binom{2n + r}{r}.$$

The truncation fixes the polynomial model class, while the neural coefficients make it data-dependent.

Parameterization and Hamiltonian Vector Field We parameterize $g_\theta \in P_{\leq r}^{\theta}$ as

$$g_\theta(z, t) = \sum_{|\boldsymbol{\alpha}|+|\boldsymbol{\beta}| \leq r} c_{\boldsymbol{\alpha},\boldsymbol{\beta}}(t; \theta)\, q^{\boldsymbol{\alpha}} p^{\boldsymbol{\beta}},$$

where the coefficients $c_{\boldsymbol{\alpha},\boldsymbol{\beta}}$ are neural network outputs depending on t (and optionally other inputs). Since the z-dependence appears only through monomials $q^{\boldsymbol{\alpha}} p^{\boldsymbol{\beta}}$, spatial derivatives act exactly as in the classical polynomial case. The induced Hamiltonian dynamics $(\dot{\mathbf{q}}, \dot{\mathbf{p}}) = J\nabla_z g_\theta$ read, for $i = 1, \ldots, n$,

$$\dot{q}_i = \frac{\partial g_\theta}{\partial p_i} = \sum_{\boldsymbol{\alpha},\boldsymbol{\beta}} c_{\boldsymbol{\alpha},\boldsymbol{\beta}}(t; \theta)\, \beta_i\, q^{\boldsymbol{\alpha}} p^{\boldsymbol{\beta} - \mathbf{e}_i}, \quad \dot{p}_i = -\frac{\partial g_\theta}{\partial q_i} = -\sum_{\boldsymbol{\alpha},\boldsymbol{\beta}} c_{\boldsymbol{\alpha},\boldsymbol{\beta}}(t; \theta)\, \alpha_i\, q^{\boldsymbol{\alpha} - \mathbf{e}_i} p^{\boldsymbol{\beta}}.$$

A.2. Sympleticity in Neural–Poisson Lie Group Walk

To compare results of created with our structure fairly with other Hamiltonian Neural Network model, our model must keep simplecticity over all Poisson-induced flow, which provides physical consistency for the Hamiltonian evolution. For any ordered dictionary $\mathbf{g}$, let $\mathscr{T}_{r,\mathcal{K}}$ be the inclusive prefixes of the Neural–Poisson Lie Group Walk, which preserves the following three distinctive symplectic properties:

$$\underbrace{\mathcal{W}_{\mathbf{g}}^{*}\omega = \omega}_{\textit{structure}}, \quad \underbrace{H(\mathcal{W}_{\mathbf{g}}z) = H(z)}_{\textit{energy}}, \quad \underbrace{\det D\mathcal{W}_{\mathbf{g}} = 1}_{\textit{volume}}, \quad \forall m \le M,\ z \in \mathcal{M}.$$

The discrete trajectory $z_m = \mathcal{W}_{\mathbf{g}}(m)z_0$ preserves the symplectic form, the Liouville measure $\mu_\omega = \omega^{\wedge n}/n!$, and the energy exactly at every step.

This equation implies that physical consistencies are preserved in the Neural–Poisson integrator, such as the energies or the overall volume at the phase-space state for the whole Hamiltonian evolution done by Neural–Poisson Lie Group Walk.

Associativity and Lie Group The weightly point of our structure is associativity for parallel scan. Let $\mathfrak{X}(\mathcal{M})$ denote the Lie algebra of smooth vector fields on the symplectic manifold with standard Lie bracket.

$$t_{r,K} := \operatorname{span}\left\{ L_{g,\le r} \;\middle|\; g \in \mathcal{G}^{(\le K)} \right\} \subset \mathfrak{X}(\mathcal{M}).$$

By construction, $t_{r,K}$ is finite linear span due to cutoff r and k, and thus finite dimensional. From Poisson-Jacobi identity, and from the fact that $\mathcal{G}^{(\le K)}$ is closed under the truncated Poisson bracket, the closure relation $[L_{g_1,\le r},\, L_{g_2,\le r}] = L_{\{g_1,g_2\},\le r} \in t_{r,K}$. Thus $t_{r,K}$ is a Lie subalgebra of $\mathfrak{X}(\mathcal{M})$. This is the keypoint for associativity since $\exp(t_{r,K})$ is an embedded Lie subgroup by the standard Lie theory (Hilgert and Neeb, 2012), and it's sufficient to show that

$$\mathscr{T}_{r,K} := \left\{ \Phi_{\epsilon,g} := \exp\big(\epsilon \mathcal{L}_{g,\le r}\big) = \sum_{j=0}^{r} \tfrac{\epsilon^j}{j!}\mathcal{L}_{g,\le r}^{j} \;\middle|\; g \in \mathcal{G}^{(\le K)} \right\}$$

is the Lie subgroup, and since the Lie subgroup inherits the original Lie group's associativity property for the combining operator $\circ$, follows as

$$\textbf{(Lie Group Associativity):} \quad \big(\Phi_{\epsilon,g_m} \circ \Phi_{\epsilon,g_{m-1}}\big) \circ \Phi_{\epsilon,g_{m-2}} = \Phi_{\epsilon,g_m} \circ \big(\Phi_{\epsilon,g_{m-1}} \circ \Phi_{\epsilon,g_{m-2}}\big).$$

Note that we confine the hypothesis space $\mathcal{H}$ to finite-degree polynomials, forcing all Hamiltonian generators and their combinations to remain closed in a finite-dimensional algebra.

This $\mathscr{T}_{r,K}$'s associativity property allows to rearrange the order of operations arbitrarily while preserving all symplectic invariants, ensuring that any regrouping of flow maps leaves the symplecticity unchanged. The parallel scanning algorithm can be used thanks to this property, since it assures the physical consistency while calculating the compositions in arbitrary order.

Appendix B. Examples and Experiments

B.1. DAG examples

The toy example provided underneath with parameter $r = 2, n = 2$, the directed acyclic graph (DAG) grows repeatedly by pairing elementary polynomials with Poisson bracket and product. The canceled representations indicate eliminated terms due to the rules according to algebraic equivalence; commutativity, antisymmetry and Jacobi Identity.

Example: Polynomial expansions in DAG construction ($r = 2$, $n = 2$)

$$\mathcal{G}^{(0)} : \Big\{c_1 q_1 + c_2 q_2 + c_3 p_1 + c_4 q_1 q_2 + c_5 q_1 p_2 + c_6 q_2 p_1 + c_7 q_2 p_2 + +c_8 p_1 p_2 + c_9 p_2^2, \dots\Big\} := P_{\leq r}^{\theta}$$

$$\mathcal{G}^{(1)} : \Big\{c_{12} q_1 \star_\theta q_2, \cancel{c_{21} q_2 \star_\theta q_1}, c_{23}\{q_2, p_1\}, c_{24} q_2^2 \star_\theta p_2, c_{34}\{q_1^2, q_1 q_2\}, \dots\Big\}, \quad \textbf{Equivalence in Product}$$

$$\mathcal{G}^{(2)} : \Big\{\cancel{c_{123}\{\{q_1, q_2\}, p_1\}}, c_{231}\{\{q_2, p_1\}, q_1\}, c_{312}\{\{p_1, q_1\}, p_2\}, \dots\Big\}, \quad \textbf{Equivalence in Jacobi Identity}$$

Formally, given arbitrary neural vector coefficients $\mathbf{c}_k(z, t; \theta), \mathbf{c}_l(z, t; \theta') \in \mathbb{R}^d$, the 2nd-order and 3rd-order coefficients $\mathbf{c}_{kl} \in \mathbb{R}^{d\times d}, \mathbf{c}_{klu} \in \mathbb{R}^{d\times d\times d}$ can be defined as:

$$\mathbf{c}_{kl}(z, t; \theta, \theta') = \mathbf{c}_k(z, t; \theta)\mathbf{c}_l(z, t; \theta')^\top - \mathbf{c}_l(z, t; \theta')\mathbf{c}_k(z, t; \theta)^\top,$$

$$\mathbf{c}_{klu}(z, t; \theta, \theta', \theta'') = \mathbf{c}_{kl}(z, t; \theta, \theta')\mathbf{c}_u(z, t; \theta'')^\top - \mathbf{c}_u(z, t; \theta'')\mathbf{c}_{kl}(z, t; \theta, \theta')^\top.$$

As can be seen, neural coefficients that begin as simple first- and second-order monomials are recursively blended through successive products and Poisson brackets into progressively richer higher-order features, so that by depth k the graph already encodes all mixed interactions of order $k + 1$ while automatically pruning algebraically redundant terms.

B.2. Parallel Scan

The accompanying figure demonstrates the parallel scanning procedure more explicitly. In the figure, each leaf node depicted in different color represents a discretized Hamiltonian generator taken from an ordered Hamiltonian dictionary. Each generator is shown only in part for clarity.

In the $V_m^{(0)}$, each generator is paired with its adjacent generator through the defined scanning operator. For instance, the red and blue blocks in $V_m^{(1)}$ are stacked together after the operation. This merging continues layer by layer, effectively reducing the computational complexity from $\mathcal{O}(M)$ to $\mathcal{O}(\log M)$, halving the nodes per one-layer advance. By running the parallel scanning algorithm, the Hamiltonian flow is computed progressively, with each layer contributing directly to its construction in real time.

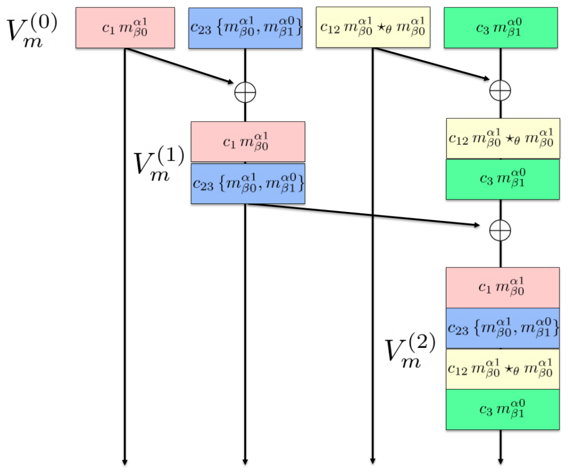
$V_m^{(0)}$
$c_1\, m_{\beta 0}^{\alpha 1}$
$c_{23}\,\{m_{\beta 0}^{\alpha 1}, m_{\beta 1}^{\alpha 0}\}$
$c_{12}\, m_{\beta 0}^{\alpha 1} \star_\theta m_{\beta 0}^{\alpha 1}$
$c_3\, m_{\beta 1}^{\alpha 0}$
$V_m^{(1)}$
$V_m^{(2)}$

Figure 2: Schematic Diagram of Parallel Scanning.

Time step	**100**	**250**	**500**	**1000**	**2000**	**5000**
(A) Training Runtime						
Euler	1.7	4.3	8.4	17.6	34.5	86.0
Störmer–Verlet	1.8	4.4	9.0	17.7	34.2	87.4
Forest–Ruth 4th	1.7	4.6	8.6	18.3	36.7	93.8
Gauss–Legendre 4th	2.5	5.8	12.1	24.4	48.8	145.7
Ours (scan)	**1.0**	**1.1**	**1.0**	**1.3**	**1.3**	**1.8**
(B) Inference Runtime						
Euler	7.0	19.2	38.1	77.5	159.8	–
Störmer–Verlet	8.2	20.5	44.2	90.1	179.8	–
Forest–Ruth 4th	14.5	23.4	45.8	89.5	184.8	–
Gauss–Legendre 4th	16.4	26.0	47.3	94.7	199.7	–
Ours (scan)	–	**1.9**	**1.2**	**1.5**	**1.5**	–

Table 2: Runtime (seconds) across timesteps for training (top) and inference (bottom).

B.3. Experimental details

We evaluate the computational efficiency of our method on a canonical harmonic oscillator system with

$$H(q,p) = \frac{p^2}{2m} + \tfrac{1}{2}m\omega^2 q^2$$

a standard benchmark for testing Hamiltonian dynamics. All models are trained with $M = 5000$ steps and evaluated on inference trajectories up to $M = 2000$ timesteps. For a fair comparison, we implement classical symplectic integrators as baselines, and report both training and inference runtime across increasing timesteps from $M = 100$ to $M = 5000$. Table 2 shows the runtime spent for each timestep scale.