Can A.I. understand Hamiltonian mechanics?

arXiv: 2410.20951

Tae-Geun Kim Seong Chan Park

Yonsei University

November 4, 2024

the 21st Saga-Yonsei Joint Workshop

Hamiltonian Mechanics

Figure 1: Sir William Rowan Hamilton

- In 1833, Sir William Rowan Hamilton introduces a reformulation of Lagrangian mechanics - *Hamiltonian mechanics* [W. R. Hamilton, PD Hardy (1833)]
- With $p = \frac{\partial L}{\partial \dot{q}}$, Hamiltonian is defined by the *Legendre transform*

$$H(q,p) = p\dot{q} - L$$

Lagrangian Mechanics

On configuration space $(q, \dot{q})$

$$L = T - V$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} = 0$$

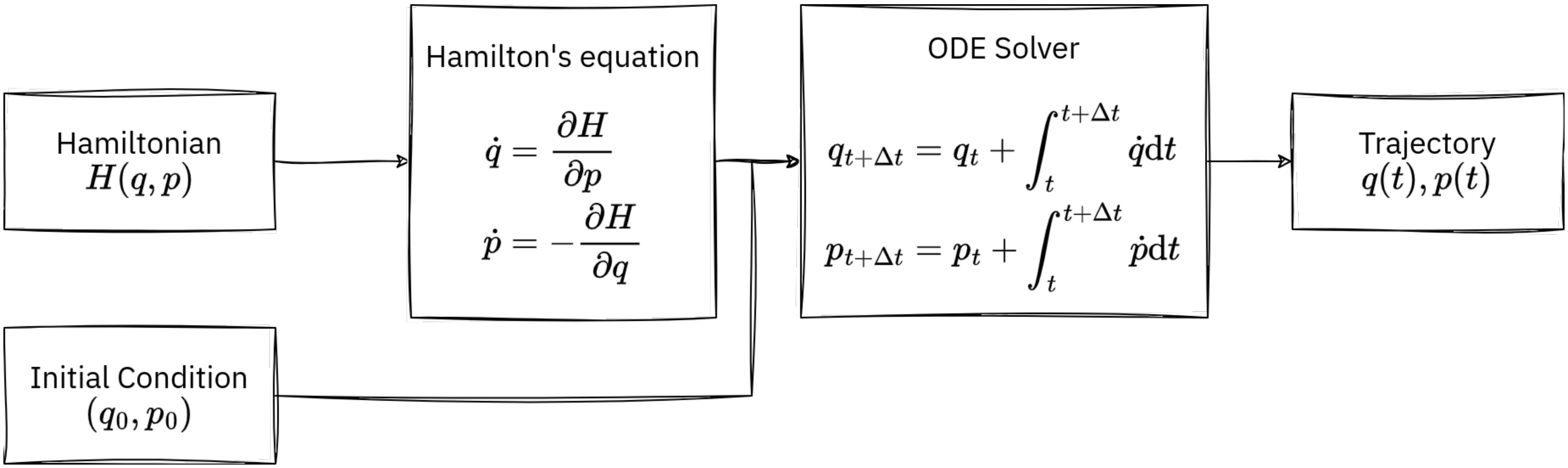
Hamiltonian Mechanics

On phase space (q, p)

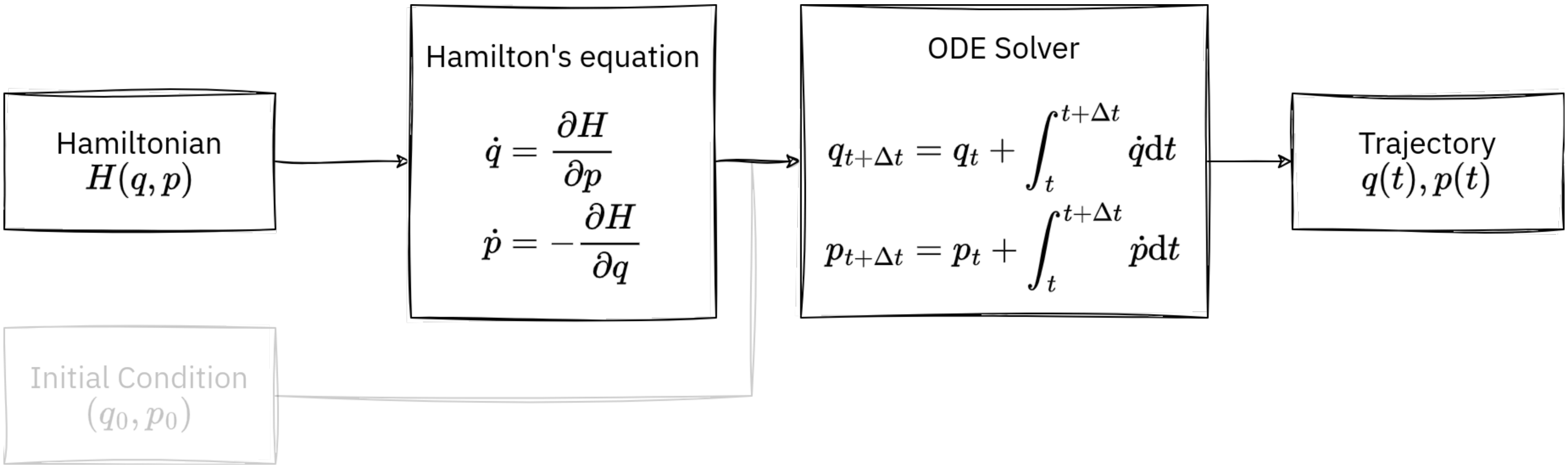
$$H = p\dot{q} - L$$

$$\dot{q} = \frac{\partial H}{\partial p}, \quad \dot{p} = -\frac{\partial H}{\partial q}$$

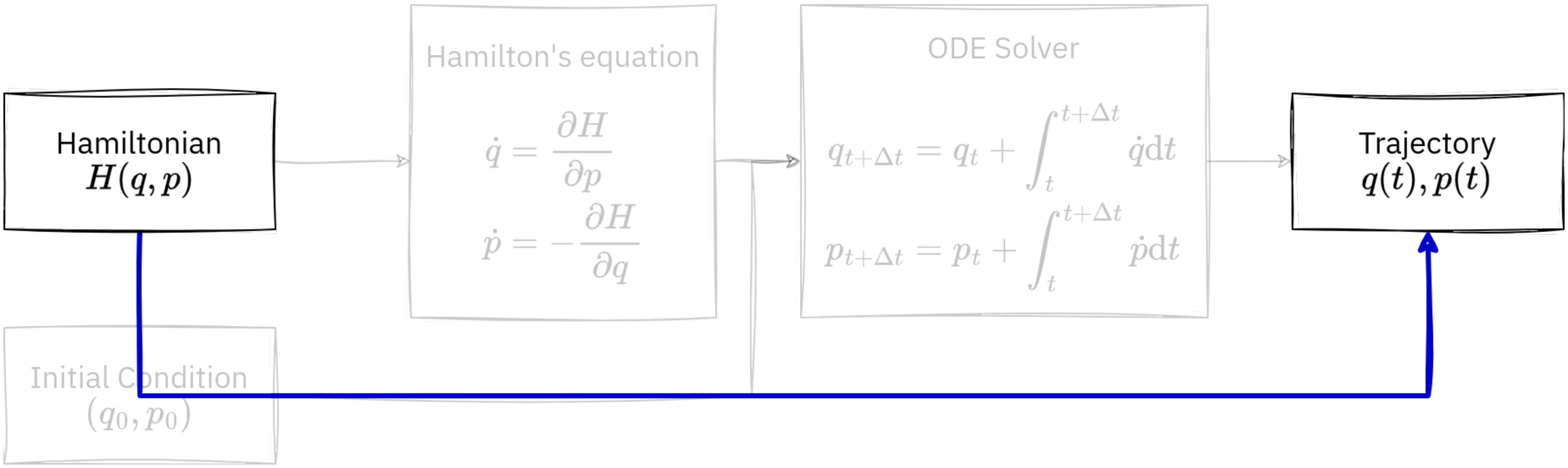
How to solve Hamilton's equation?



How to solve Hamilton's equation?



How to solve Hamilton's equation?



Q. Can A.I. do this process?

Neural Network

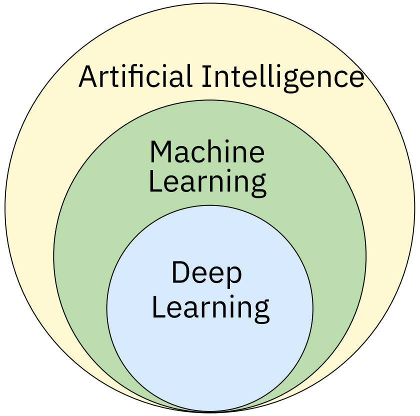


Figure 2: A.I. Hierarchy

- In this work, Artificial Intelligence refers to *(Artificial) Neural Networks* (NN)
- Neural Network finds $\hat{f}$ that best maps input to output:

$$\text{Training Data } \mathscr{D} = \{(x_i, y_i)\}_{i=1}^{n} \Longrightarrow \text{Learn } \hat{f} \text{ such that } y_i \approx \hat{f}(x_i)$$

- The convergence is guaranteed by *Universal Approximation Theorem* (UAT)

[Lu et al., NeurIPS (2017); G. Cybenko, MCSS (1989)]

Theorem 1 (Universal Approximation Theorem for Width-Bounded ReLU Networks). *For any Lebesgue-integrable function* $f : \mathbb{R}^n \to \mathbb{R}$ *and any* $\epsilon > 0$, *there exists a fully-connected ReLU network* $\mathscr{A}$ *with width* $d_m \leq n + 4$, *such that the function* $F_{\mathscr{A}}$ *represented by this network satisfies*

$$\int_{\mathbb{R}^n} |f(x) - F_{\mathscr{A}}(x)| \mathrm{d}x < \epsilon. \tag{3}$$

Neural Network

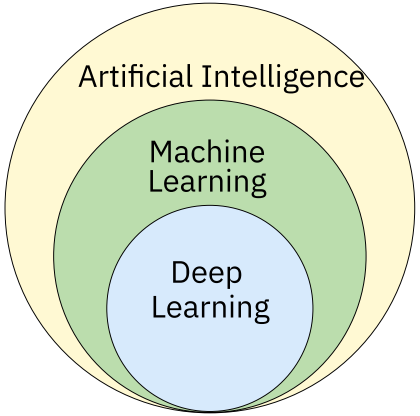


Figure 2: A.I. Hierarchy

- In this work, Artificial Intelligence refers to *(Artificial) Neural Networks* (NN)
- Neural Network finds $\hat{f}$ that best maps input to output:

$$\text{Training Data } \mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n} \Longrightarrow \text{Learn } \hat{f} \text{ such that } y_i \approx \hat{f}(x_i)$$

- The convergence is guaranteed by *Universal Approximation Theorem* (UAT)

[Lu et al., NeurIPS (2017); G. Cybenko, MCSS (1989)]

Theorem 1 (Universal Approximation Theorem for Width-Bounded ReLU Networks)**.** *For any Lebesgue-integrable function $f : \mathbb{R}^n \to \mathbb{R}$ and any $\epsilon > 0$, there exists a fully-connected ReLU network $\mathscr{A}$ with width $d_m \leq n+4$, such that the function $F_{\mathscr{A}}$ represented by this network satisfies*

$$\int_{\mathbb{R}^n} |f(x) - F_{\mathscr{A}}(x)| \mathrm{d}x < \epsilon. \tag{3}$$

Q. But, how to approximate $G : H(q,p) \mapsto (q(t), p(t))$?

Operator Learning

- **Universal Approximation Theorem for Operator**

[Lu et al., Nat. Mach. Intell. (2021); Chen & Chen, IEEE Trans. Neural Netw. (1995)]

Theorem 1. *Suppose that X is a Banach space, $K_1 \subset X, K_2 \subset \mathbb{R}^d$ are two compact sets in X and $\mathbb{R}^d$, respectively, V is a compact set in $C(K_1)$. Assume that $G : V \to C(K_2)$ is a nonlinear continuous operator. Then, for any $\epsilon > 0$, there exist positive integers m, p, continuous vector functions $\boldsymbol{g} : \mathbb{R}^m \to \mathbb{R}^p$, $\boldsymbol{f} : \mathbb{R}^d \to \mathbb{R}^p$, and $x_1, x_2, \cdots, x_m \in K_1$, such that*

$$\left| G(u)(y) - \langle \underbrace{\boldsymbol{g}(u(x_1), u(x_2), \cdots, u(x_m))}_{branch}, \underbrace{\boldsymbol{f}(y)}_{trunk} \rangle \right| < \epsilon \qquad (1)$$

holds for all $u \in V$ and $y \in K_2$.

- **DeepONet** [Lu et al., Nat. Mach. Intell. (2021)]
  - Task: Approximate the operator $G : u(x) \mapsto G(u)(y)$
  - Train data
    - Discretization of input function $[u(x_1), ..., u(x_m)]$
    - Query point y
    - Output function as label $G(u)(y)$

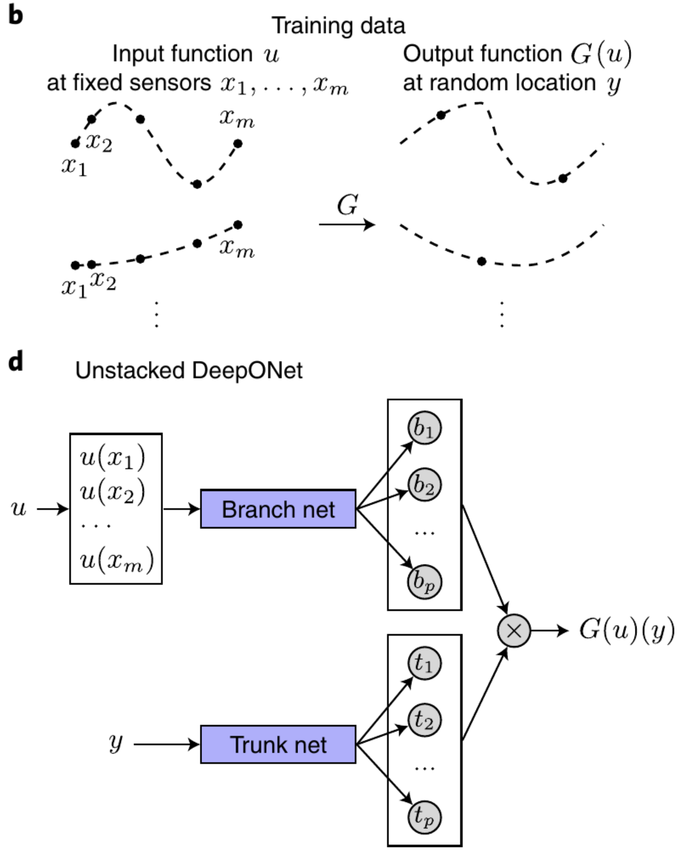


Figure 3: DeepONet structure

Example: Anti-Derivative

- **Task**: For $u \in C[0,1]$ and $y \in [0,1]$, learn the operator such that

$$G(u)(y) = \int_0^y u(x)\mathrm{d}x$$

- **Input data**
  - u: Input function from *Gaussian Random Field*

$$u(x) = \mathrm{Spline}\left[\{(x_i, \mathrm{GRF}(x_i))\}_{i=1}^m\right](x)$$

  - y: Target point

- **Label**

$$G(u_i)(y_j) = \int_0^{y_j} u_i(x)\mathrm{d}x$$

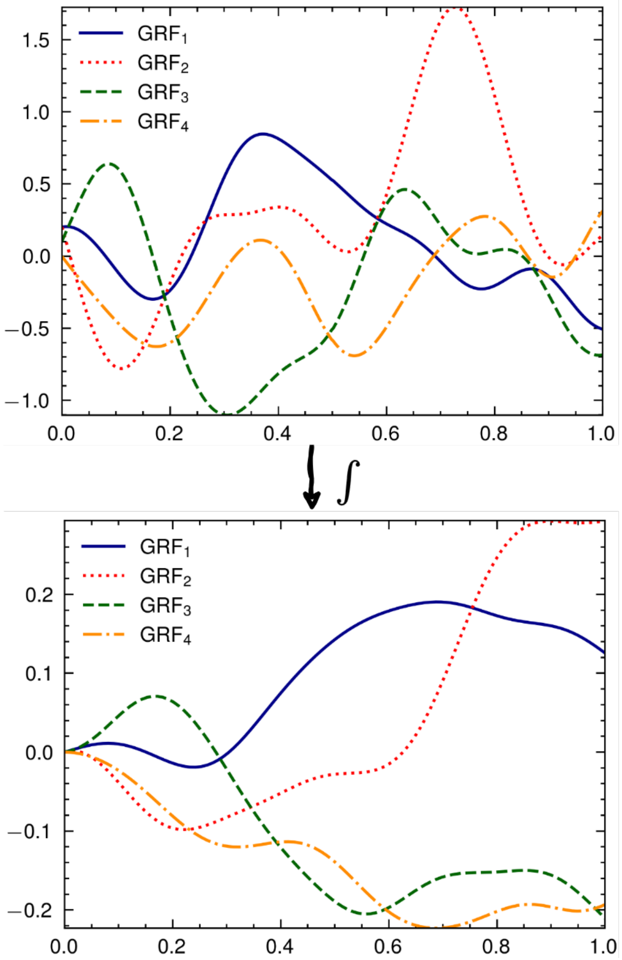
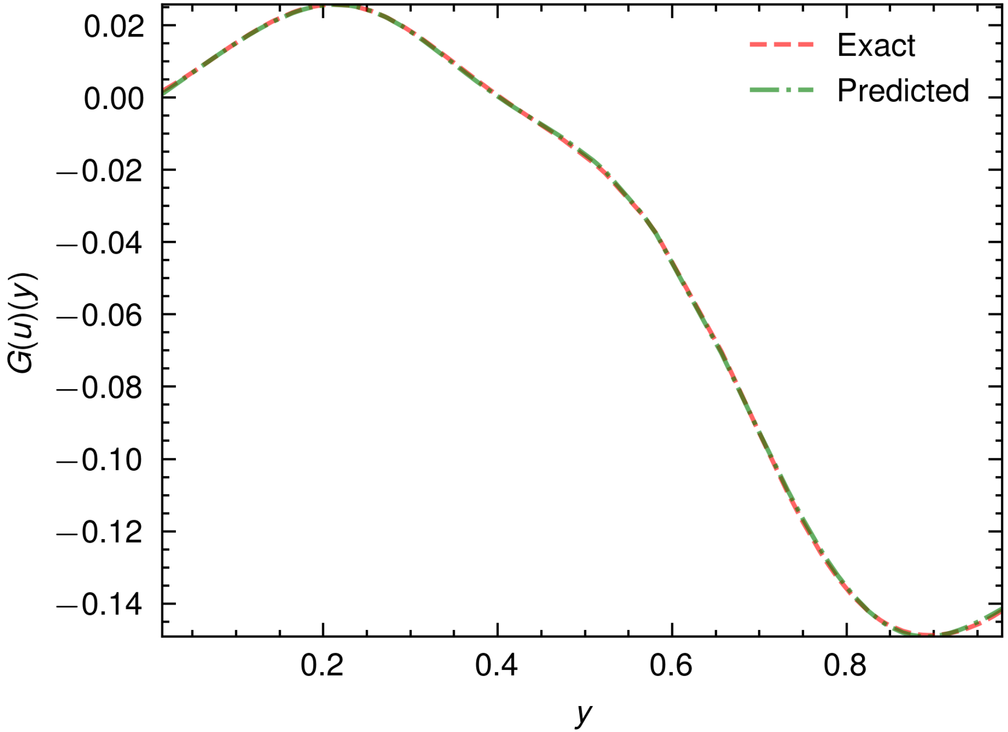


Figure 4: (Top) $u(x)$ and (Bottom) $G(u)(y)$

Example: Anti-Derivative



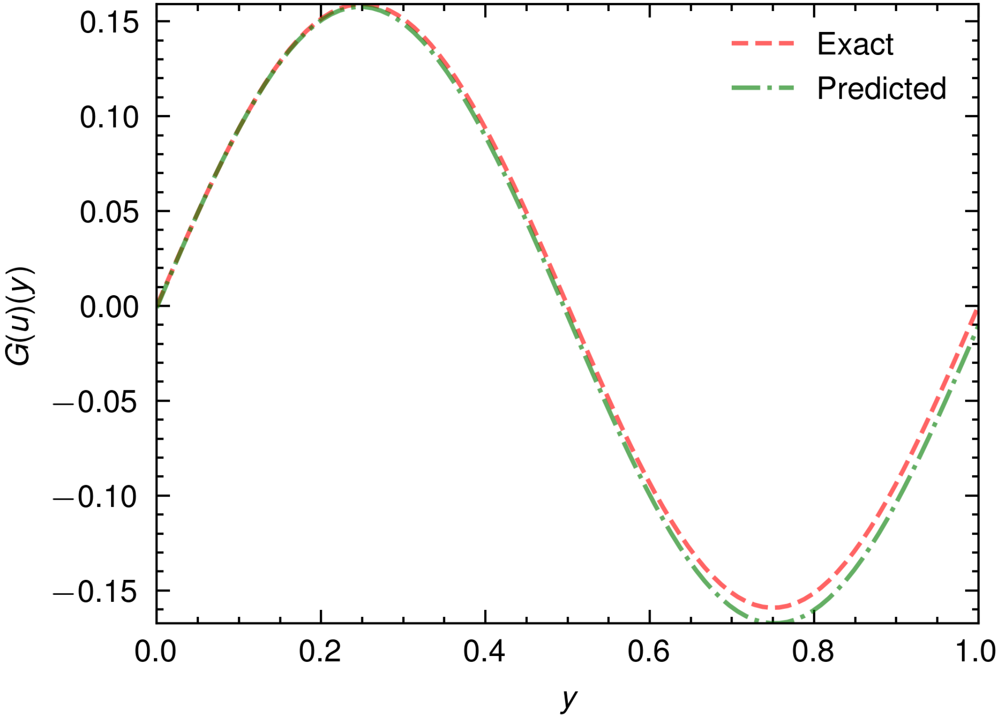
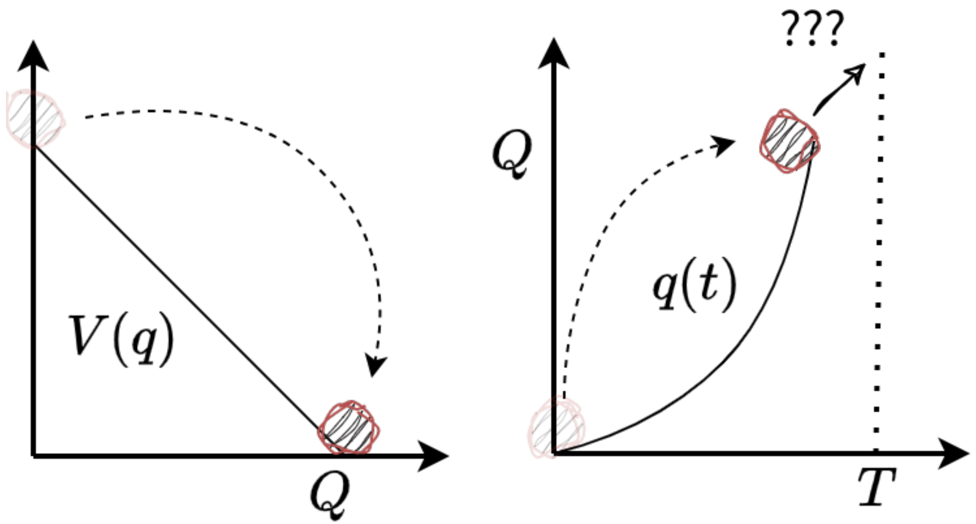


Figure 5: (Left) Test for one GRF sample and (Right) test for cosine function

Operator Learning for Hamilton's Equation

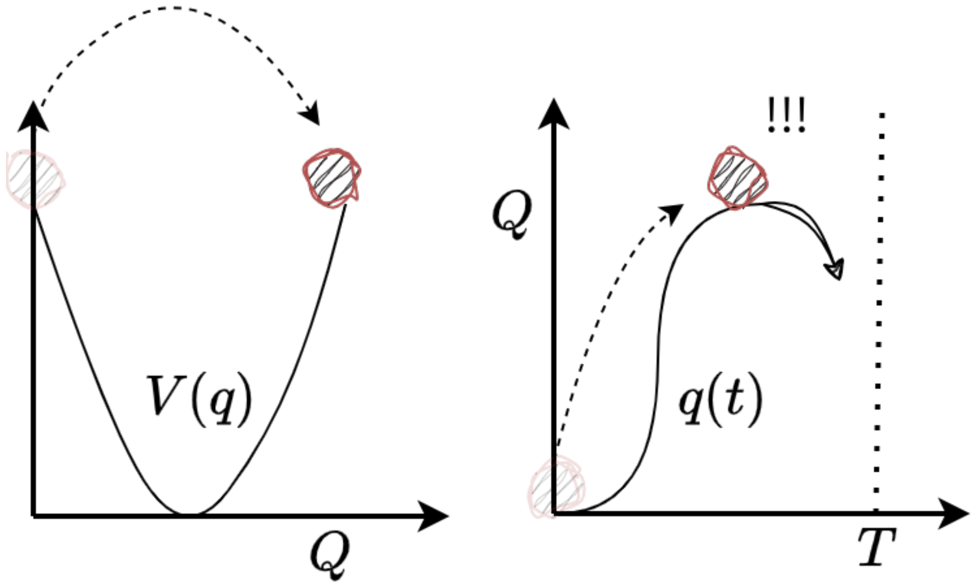
- **Task**: For $H(q,p) = \frac{p^2}{2} + V(q)$, approximate the operator $G : V(q) \mapsto (q(t), p(t))$
- **Constraints**
  - From the UAT for operator, domain & range of V and q, p should be compact



  - Simply making $V(q)$ bounded on a compact domain is insufficient
  - Even with that kind of potential, trajectories may escape the domain
  - As shown in the figure, $q(t)$ can exceed the boundary Q before time T
  - This leads to an ill-defined problem where $V(q)$ doesn't exist for $q > Q$

Operator Learning for Hamilton's Equation

- **Task**: For $H(q,p) = \frac{p^2}{2} + V(q)$, approximate the operator $G : V(q) \mapsto (q(t), p(t))$
- **Constraints**
  - From the UAT for operator, domain & range of V and q, p should be compact



- Simply making $V(q)$ bounded on a compact domain is insufficient
- Even with that kind of potential, trajectories may escape the domain
- As shown in the figure, $q(t)$ can exceed the boundary Q before time T
- This leads to an ill-defined problem where $V(q)$ doesn't exist for $q > Q$

$\Rightarrow$ **Consider "bounded" potential**

Operator Learning for Hamilton's Equation

- **Task**: For $H(q,p) = \dfrac{p^2}{2} + V(q)$, approximate the operator $G : V(q) \mapsto (q(t), p(t))$
- **Constraints**
  - From the UAT for operator, domain & range of V and q, p should be compact

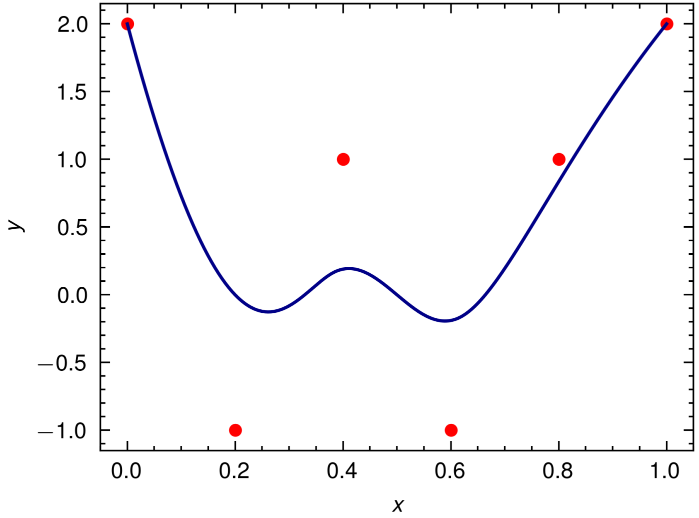
    $\Rightarrow$ **Consider "bounded" potential**
  - From the operator formulation, V must be twice continuously differentiable (C^2)
    - This ensures the existence and uniqueness of solutions
    - Required for the operator G to be well-defined

Operator Learning for Hamilton's Equation

- **Task**: For $H(q,p) = \frac{p^2}{2} + V(q)$, approximate the operator $G : V(q) \mapsto (q(t), p(t))$
- **Constraints**
  - From the UAT for operator, domain & range of V and q, p should be compact

    $\Rightarrow$ **Consider "bounded" potential**
  - From the operator formulation, V must be twice continuously differentiable (C^2)
    - This ensures the existence and uniqueness of solutions
    - Required for the operator G to be well-defined

    $\Rightarrow$ **Use "Cubic B-Spline" to generate potentials**



Data Generation

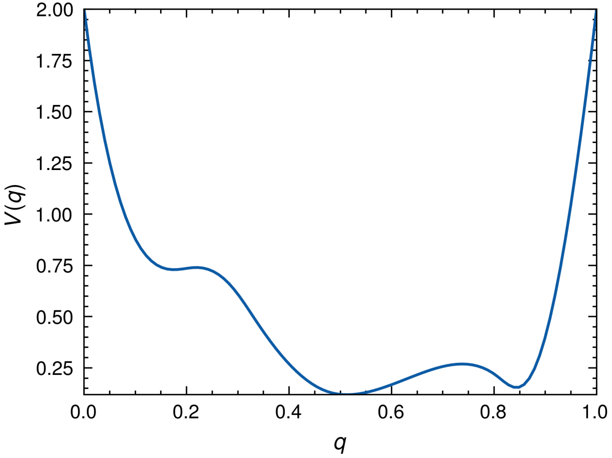


Figure 6: Potential function $V(q)$

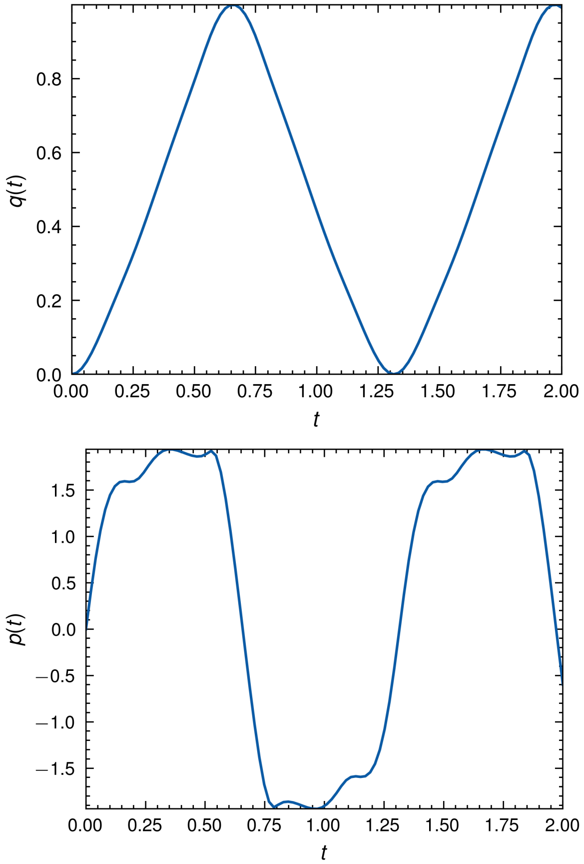


Figure 7: Trajectory $q(t)$ and $p(t)$

- Generate random potential with GRF + Cubic B-Spline
  - $V(0) = V(1) = 2$
  - $V(q) < 2$ for $0 < q < 1$
  - Standard dataset: 10,000, Extended dataset: 100,000
- Solve Hamilton's equation with *Gauss-Legendre 4th order* (GL4) to generate trajectories.
  - Initial condition: $q(0) = p(0) = 0$
  - Time: $0 \leq t \leq 2$

Models

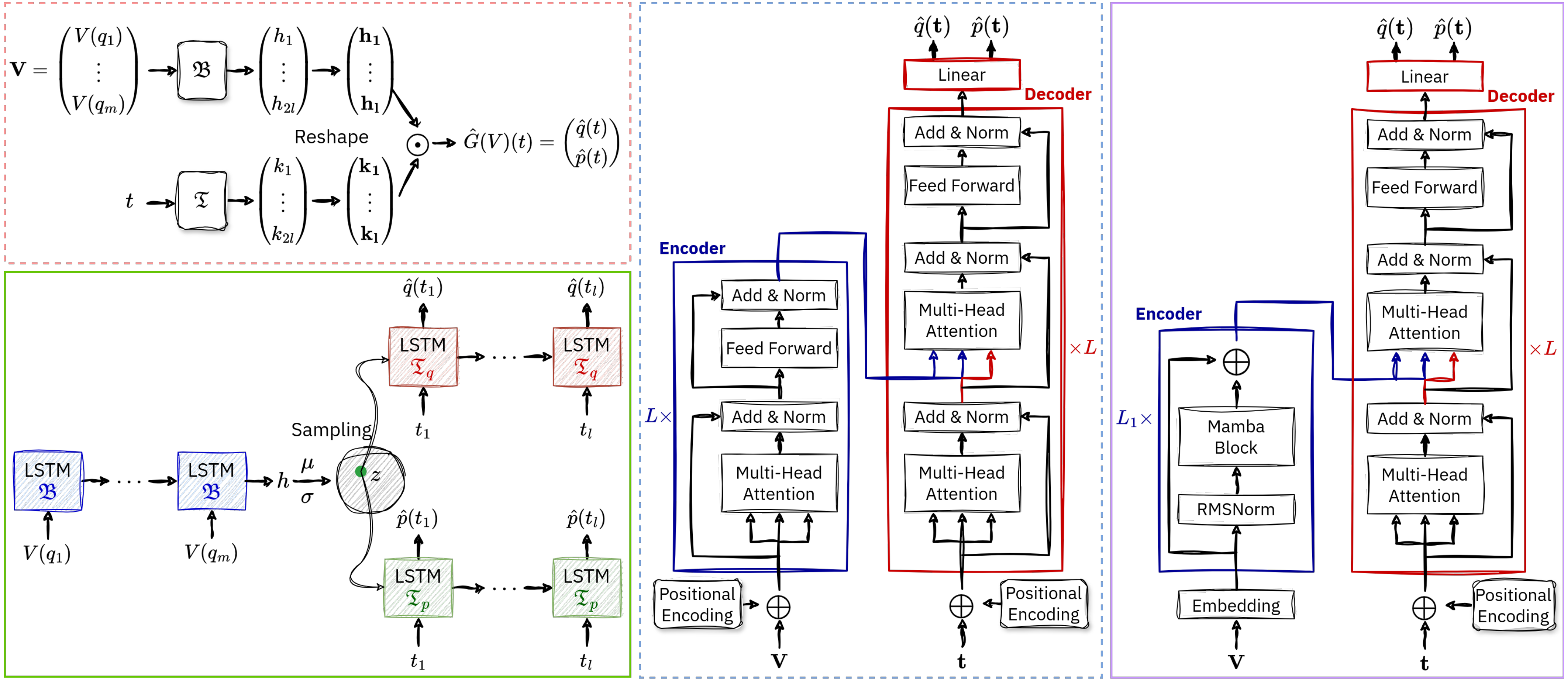


Figure 8: Models for Neural Hamilton.
(Red Dashed Box) DeepONet, (Green Solid Box) VaRONet, (Blue Dashed Box) TraONet and (Purple Solid Box) MambONet

A Result

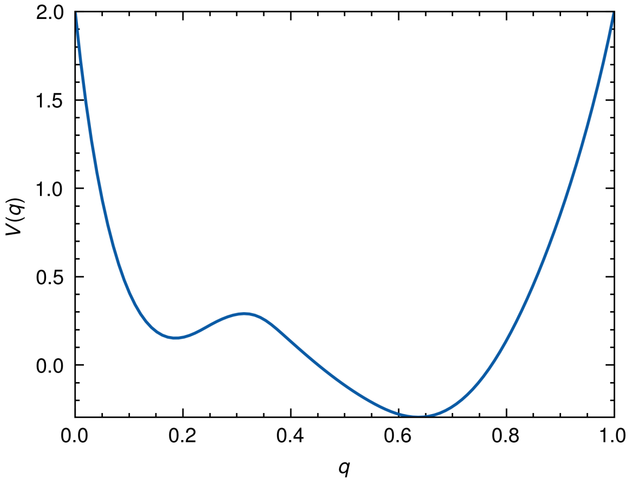


Figure 9: Potential function $V(q)$

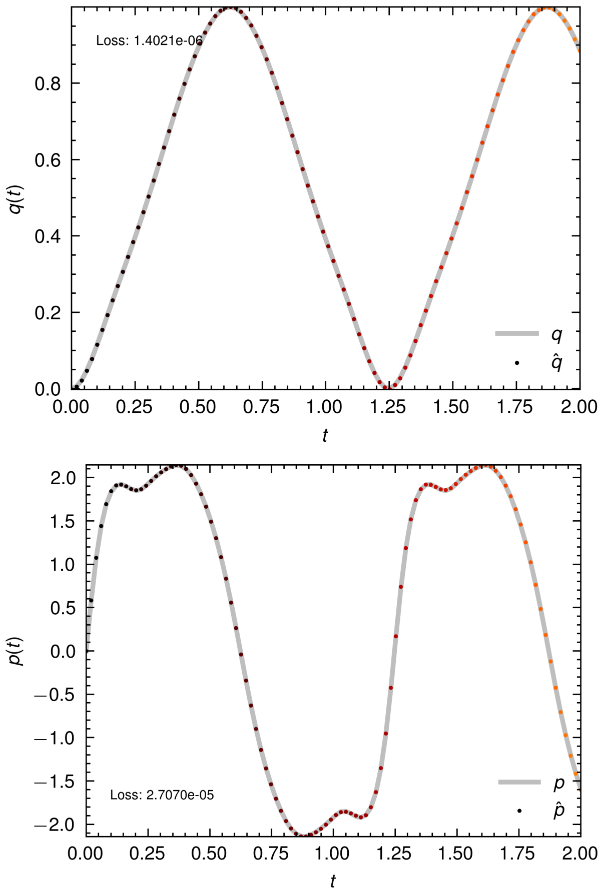


Figure 10: Trajectory $q(t)$ and $p(t)$

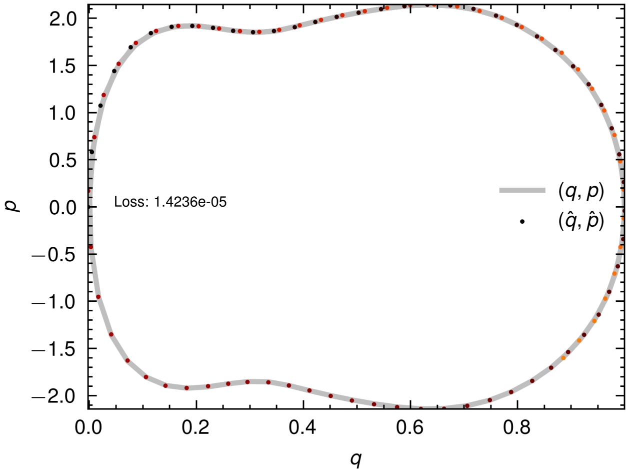
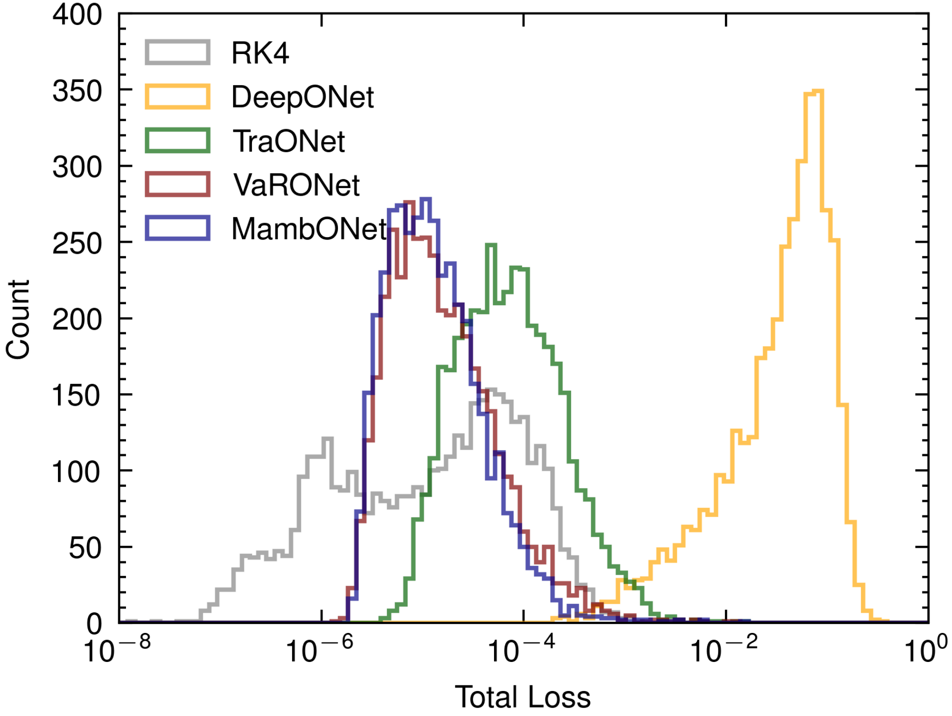


Figure 11: Phase space plot

Test Results



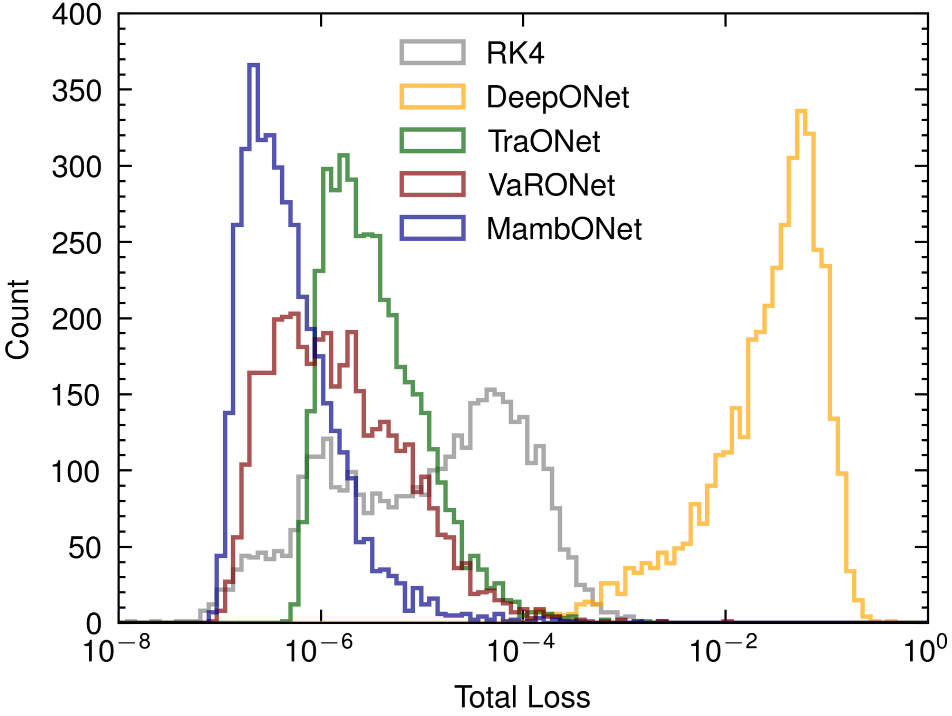


Figure 12: Histogram for total test losses for (left) standard dataset and (right) extended dataset

$$\mathcal{L}_{\text{tot}} = \frac{1}{2}(\mathcal{L}_q + \mathcal{L}_p) = \frac{1}{2N}\sum_{i=1}^{N}(\|q_i(\boldsymbol{t}) - \hat{q}_i(\boldsymbol{t}))\|^2 + \|p_i(\boldsymbol{t}) - \hat{p}_i(\boldsymbol{t}))\|^2)$$

Physically Relevant Potentials

- **Simple Harmonic Oscillator**

$$V(q) = 8\left(q - \frac{1}{2}\right)^2$$

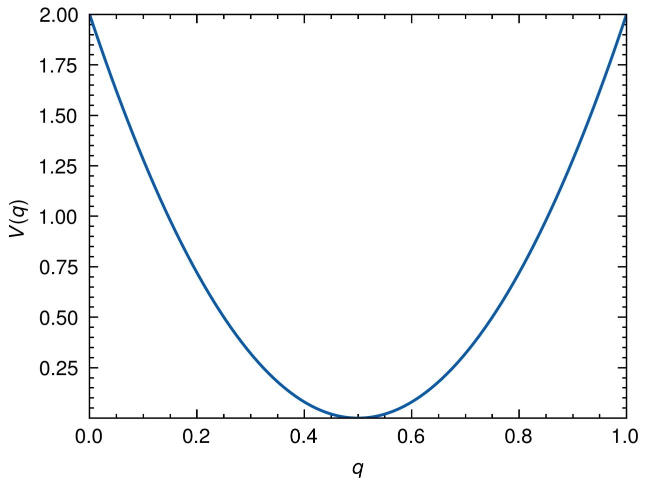


Figure 13: SHO

- **Double-Well Potential**

$$V(q) = \frac{625}{8}\left(q - \frac{1}{5}\right)^2\left(q - \frac{4}{5}\right)^2$$

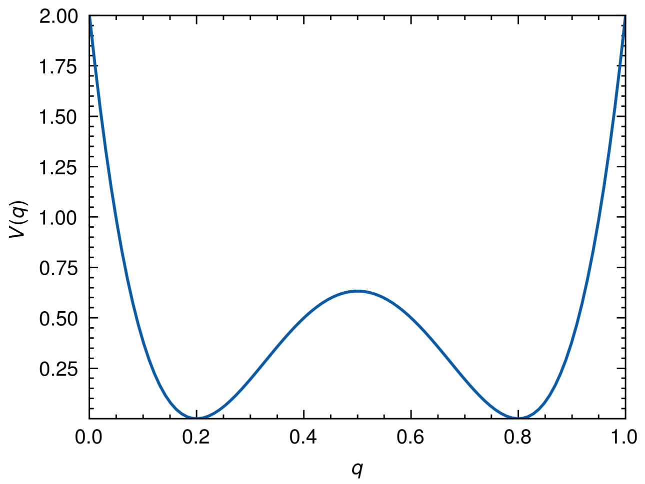


Figure 14: Double-Well

- **Morse Potential**

$$V(q) = \frac{8}{\left(\sqrt{5}-1\right)^2}\left(1 - e^{-a(q-1/3)}\right)^2$$

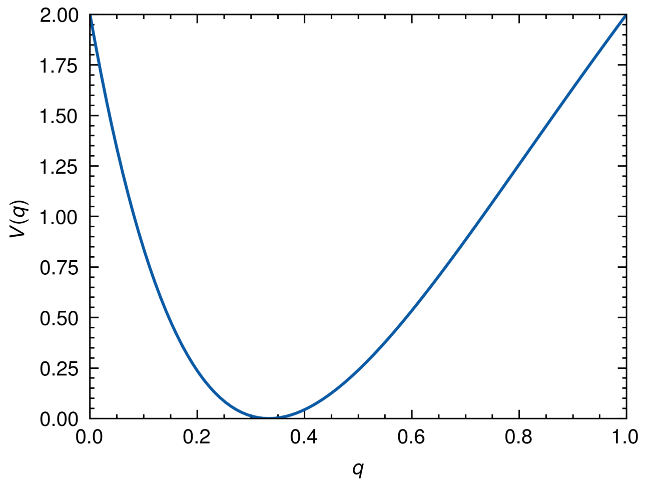


Figure 15: Morse

Physically Relevant Potentials

- **Mirrored Free-Fall**

$$V(q) = 4\left|q - \frac{1}{2}\right|$$

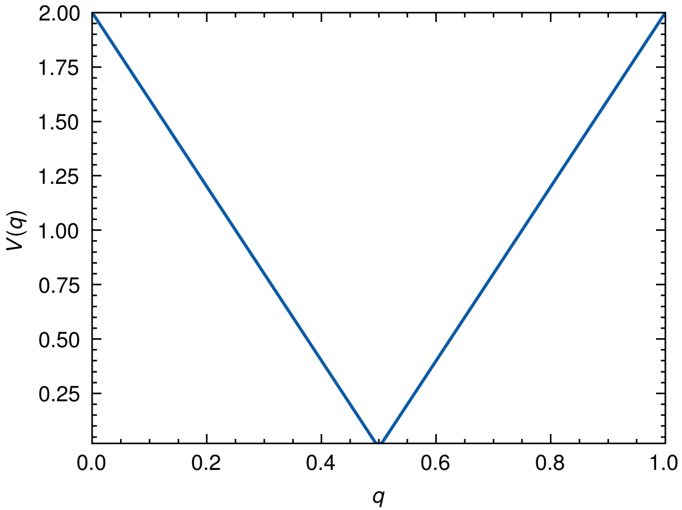


Figure 16: Mirrored Free-Fall

- **Softened Mirrored Free-Fall**

$$V(q) = \frac{4}{\coth(\frac{\alpha}{2})}\left(q - \frac{1}{2}\right)\coth\left(\alpha\left(q - \frac{1}{2}\right)\right)$$

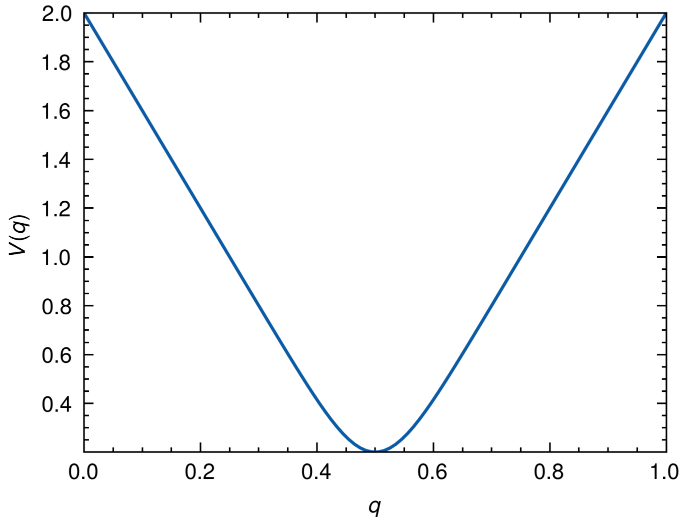


Figure 17: Softened Mirrored Free-Fall

Test Results on Physically Relevant Potentials

Table 5: Performance on physically relevant potentials (Total Loss $\mathcal{L}_{\text{tot}}$)

Model	SHO	Double Well	Morse	MFF	SMFF
RK4	3.3663×10^{-5}	1.4362×10^{-2}	2.8753×10^{-4}	1.5224×10^{-4}	4.1551×10^{-5}
DeepONet	2.4951×10^{-4}	8.9770×10^{-2}	4.7225×10^{-2}	2.5406×10^{-2}	1.5242×10^{-2}
TraONet	1.0145×10^{-6}	2.2207×10^{-6}	7.6594×10^{-6}	$\mathbf{1.0228 \times 10^{-4}}$	$\underline{3.3919 \times 10^{-5}}$
VaRONet	$\underline{1.9729 \times 10^{-7}}$	$\underline{8.8354 \times 10^{-7}}$	$\underline{2.3114 \times 10^{-6}}$	2.0465×10^{-4}	7.2018×10^{-5}
MambONet	$\mathbf{1.4197 \times 10^{-7}}$	$\mathbf{4.3837 \times 10^{-7}}$	$\mathbf{1.3451 \times 10^{-6}}$	$\underline{1.3426 \times 10^{-4}}$	$\mathbf{1.9754 \times 10^{-6}}$

Test Results on Physically Relevant Potentials

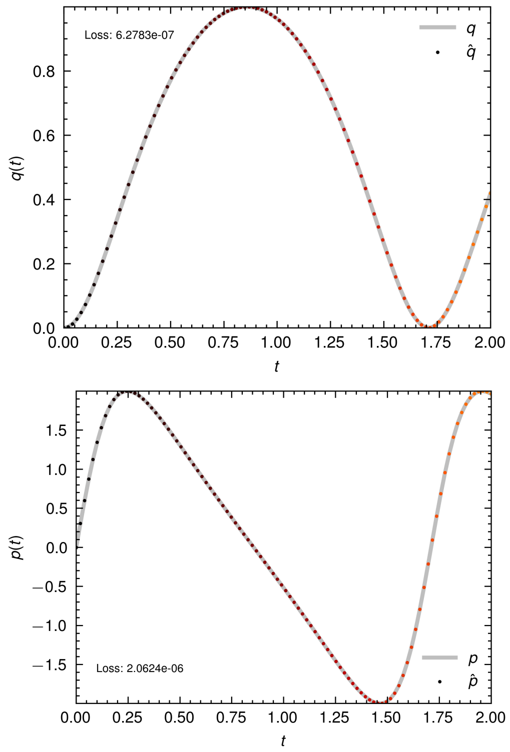


Figure 19: MambONet (Morse)

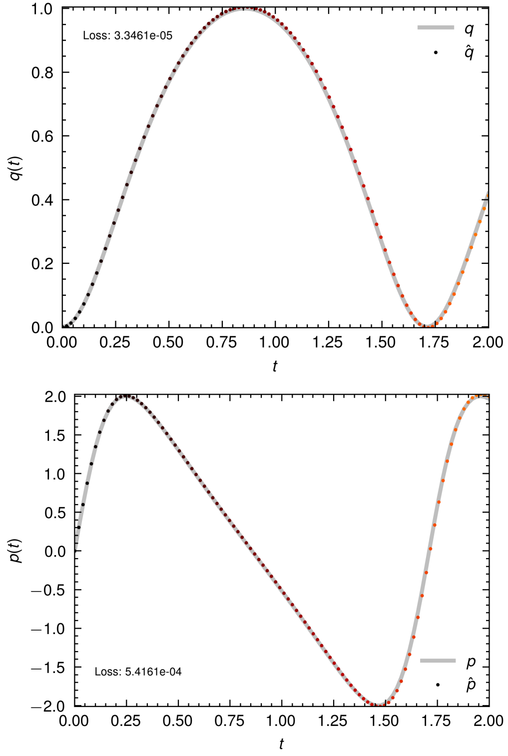


Figure 20: RK4 (Morse)

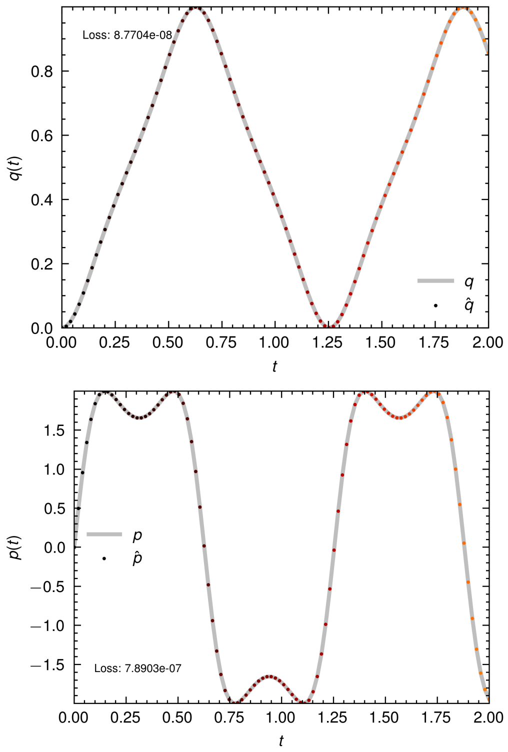


Figure 21: MambONet (Double-Well)

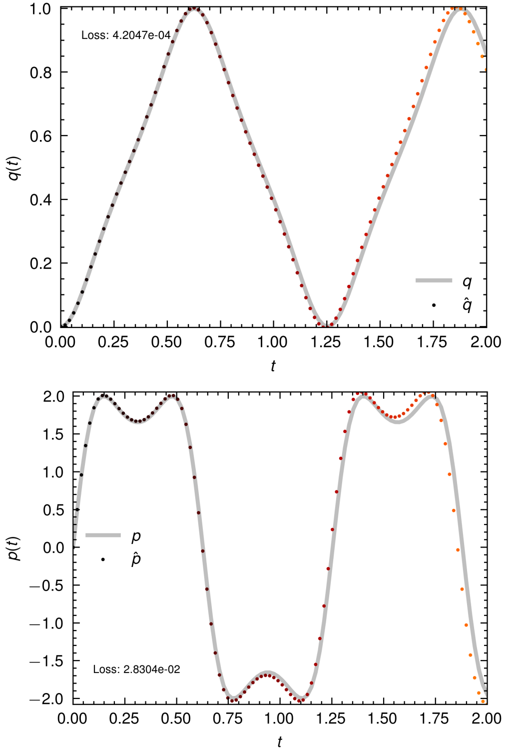


Figure 22: RK4 (Double-Well)

Extrapolation Test

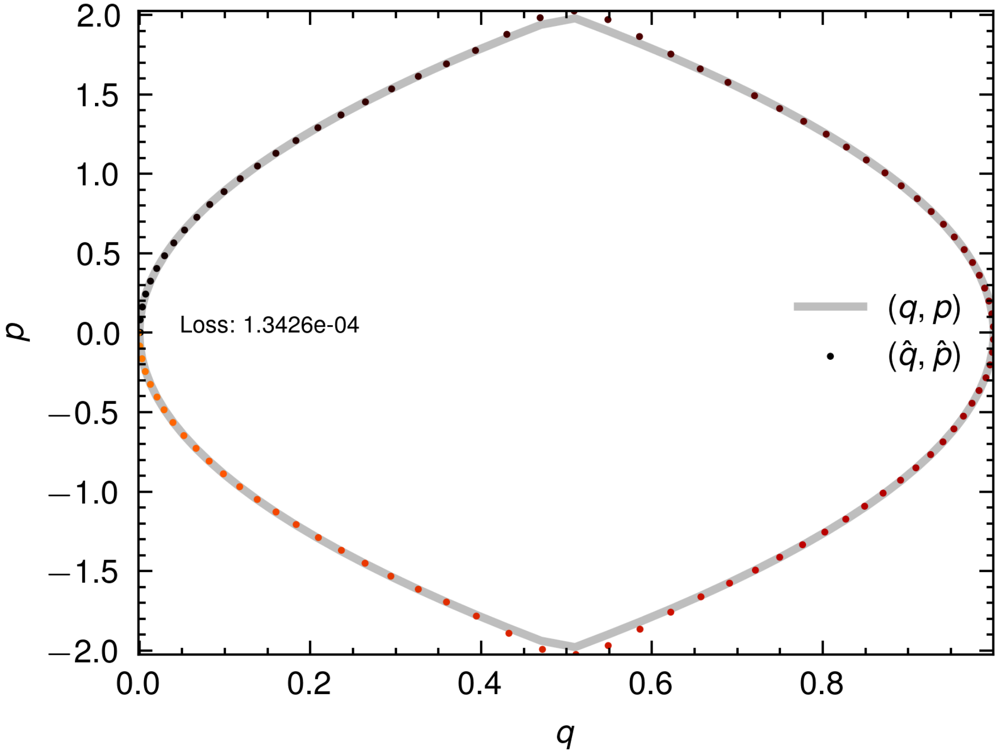


Figure 23: MambONet (Mirrored Free-Fall)

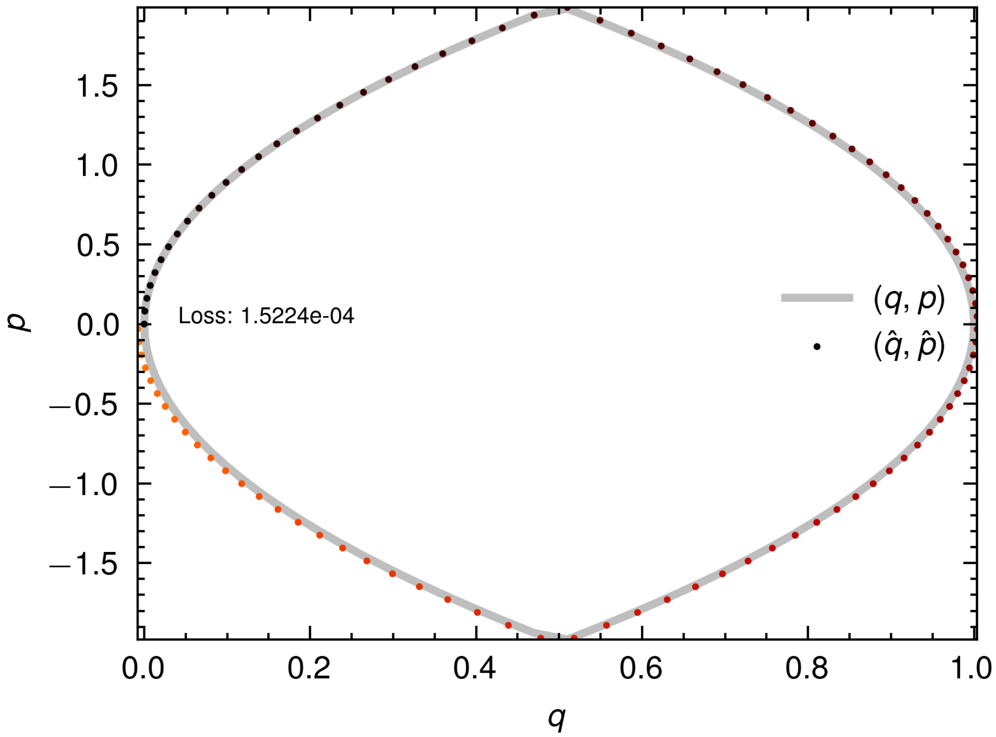


Figure 24: RK4 (Mirrored Free-Fall)

Extrapolation Test

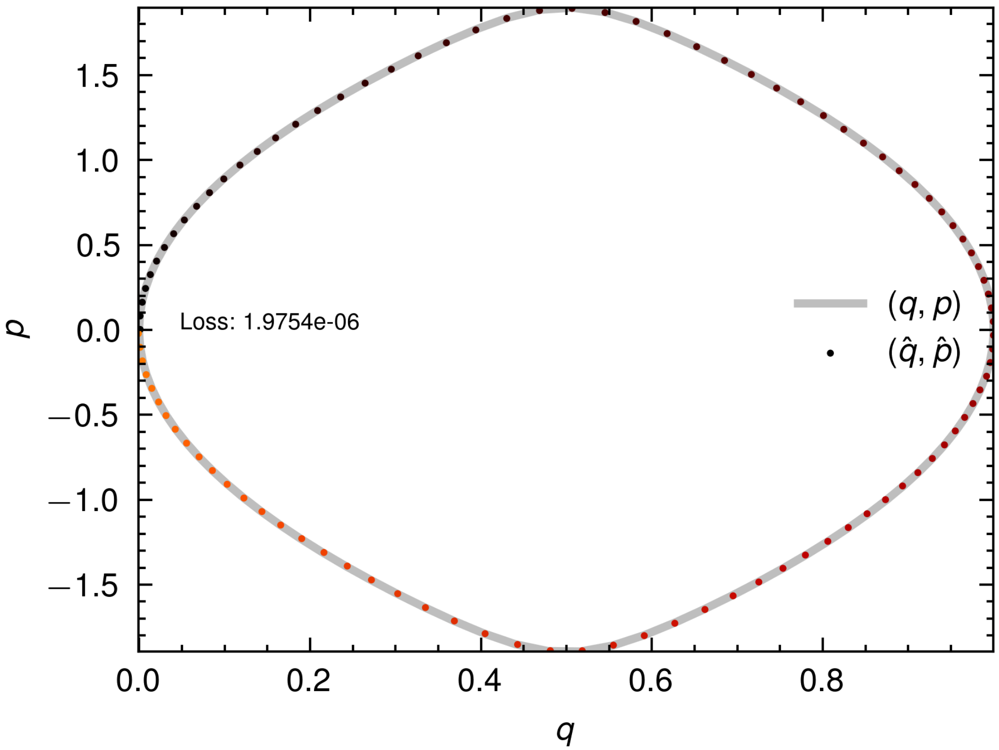


Figure 25: MambONet (Softened Mirrored Free-Fall)

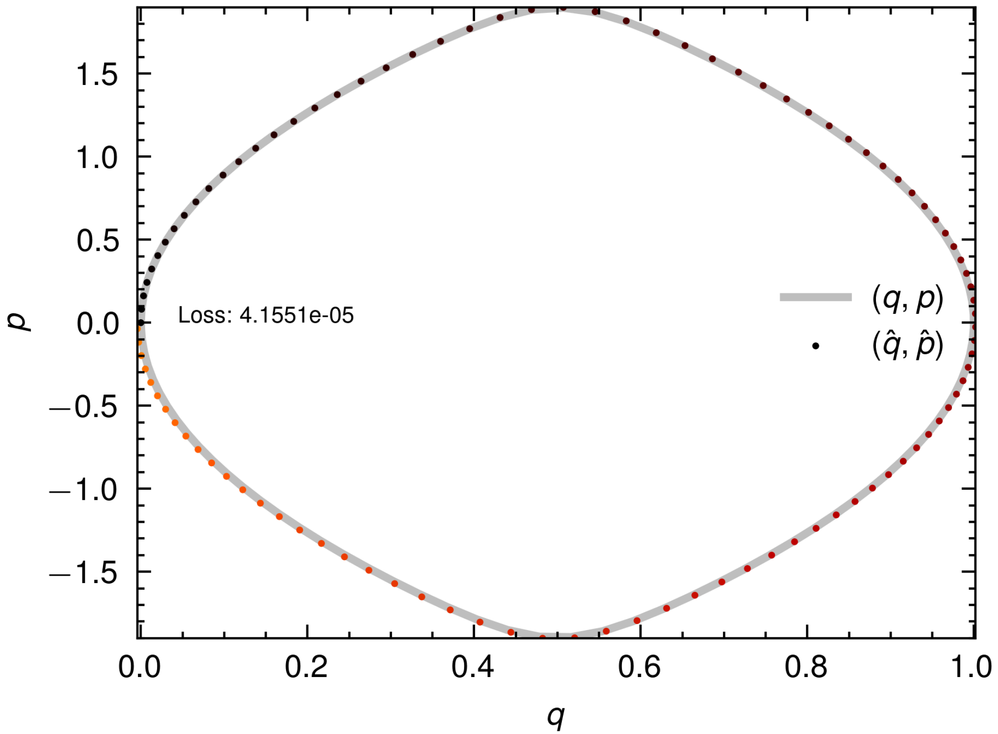
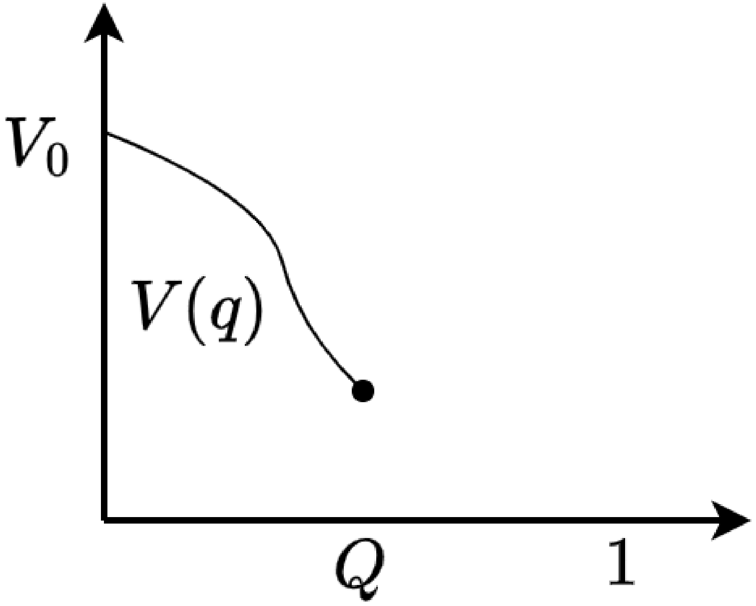


Figure 26: RK4 (Softened Mirrored Free-Fall)

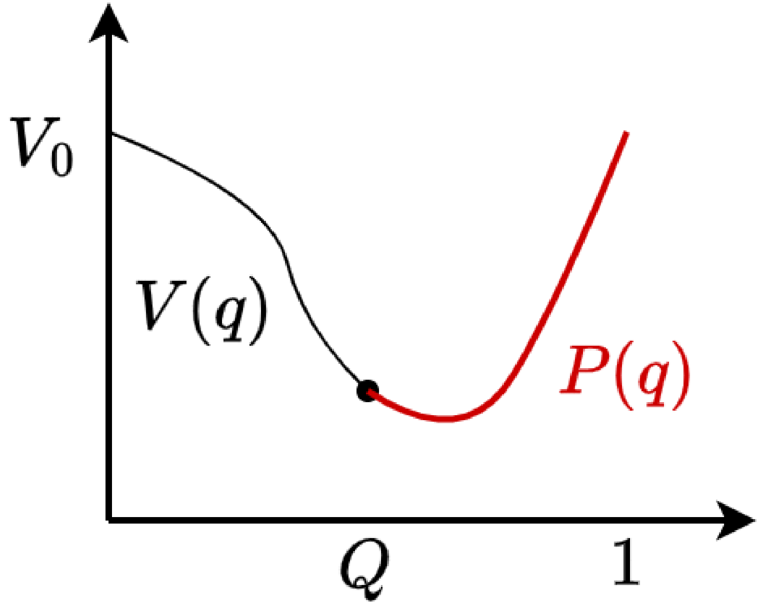
How about unbounded?

- Consider a monotonically decreasing C^2 potential $V(q)$ defined on $[0, Q]$, where $0 < Q < 1$ and $V(0) = V_0$



How about unbounded?

- Consider a monotonically decreasing C^2 potential $V(q)$ defined on $[0, Q]$, where $0 < Q < 1$ and $V(0) = V_0$



  - A new C^2 function $P(q)$ on $[Q, 1]$ such that

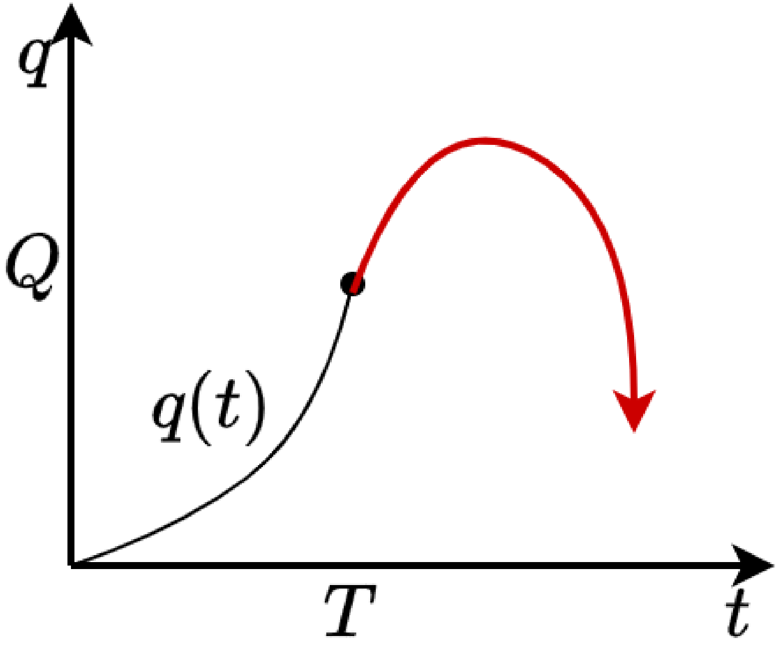
$$P(1) = V_0$$
$$P(Q) = V(Q)$$
$$P'(Q) = V'(Q)$$
$$P''(Q) = V''(Q)$$
$$P(q) < V_0 \quad \text{for} \quad Q < q < 1$$

  - Then we can define a new C^2 potential function $\tilde{V}(q)$ as

$$\tilde{V}(q) = \begin{cases} V(q) \text{ if } 0 \leq q \leq Q \\ P(q) \text{ if } Q < q \leq 1 \end{cases}$$

How about unbounded?

- Input new potential function $\tilde{V}(q)$ into the model, then we can get $q(t)$ and $p(t)$



- To extract the relevant dynamics, we determine the time T
- Since $H = \frac{p^2}{2} + V(q) = V_0$, from Hamilton's equation,

$$\frac{\mathrm{d}q}{\mathrm{d}t} = \frac{\partial H}{\partial p} = p = \sqrt{2(V_0 - V(q))}$$

$$\Rightarrow \int_0^T \mathrm{d}t = \int_0^Q \frac{\mathrm{d}q}{\sqrt{2(V_0 - V(q))}}$$

$$\Rightarrow T = \int_0^Q \frac{\mathrm{d}q}{\sqrt{2(V_0 - V(q))}}$$

- Take $q(t)$ and $p(t)$ upto time T

Example: Free-Fall

- Consider a free fall potential: $V(q) = -4(q - 0.5), \quad (0 \leq q \leq 0.5)$ [Answer: $q(t) = 2t^2, p(t) = 4t$]
  - From the previous conditions, we can find a cubic function $P(q) = 32q^3 - 48q^2 + 20q - 2$
  - Obtain the time $T = \int_0^{\frac{1}{2}} \frac{\mathrm{d}q}{\sqrt{2(2 - V(q))}} = \int_0^{\frac{1}{2}} \frac{\mathrm{d}q}{\sqrt{8q}} = 0.5$

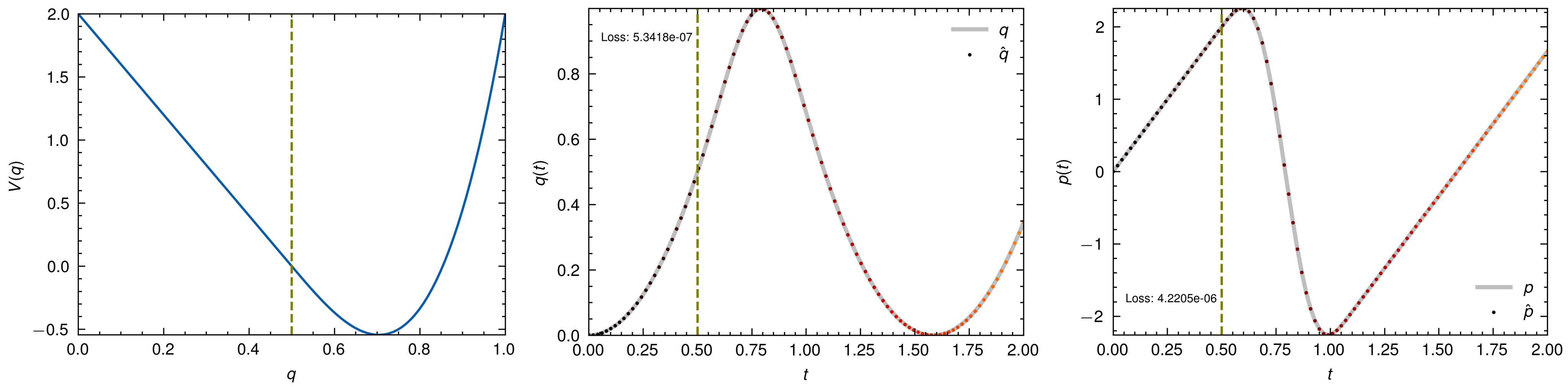


Figure 27: (Left) New potential function $\tilde{V}(q)$, (Middle) $q(t)$, (Right) $p(t)$; Olive dashed line marks the relevant area.

Large Neural Hamilton

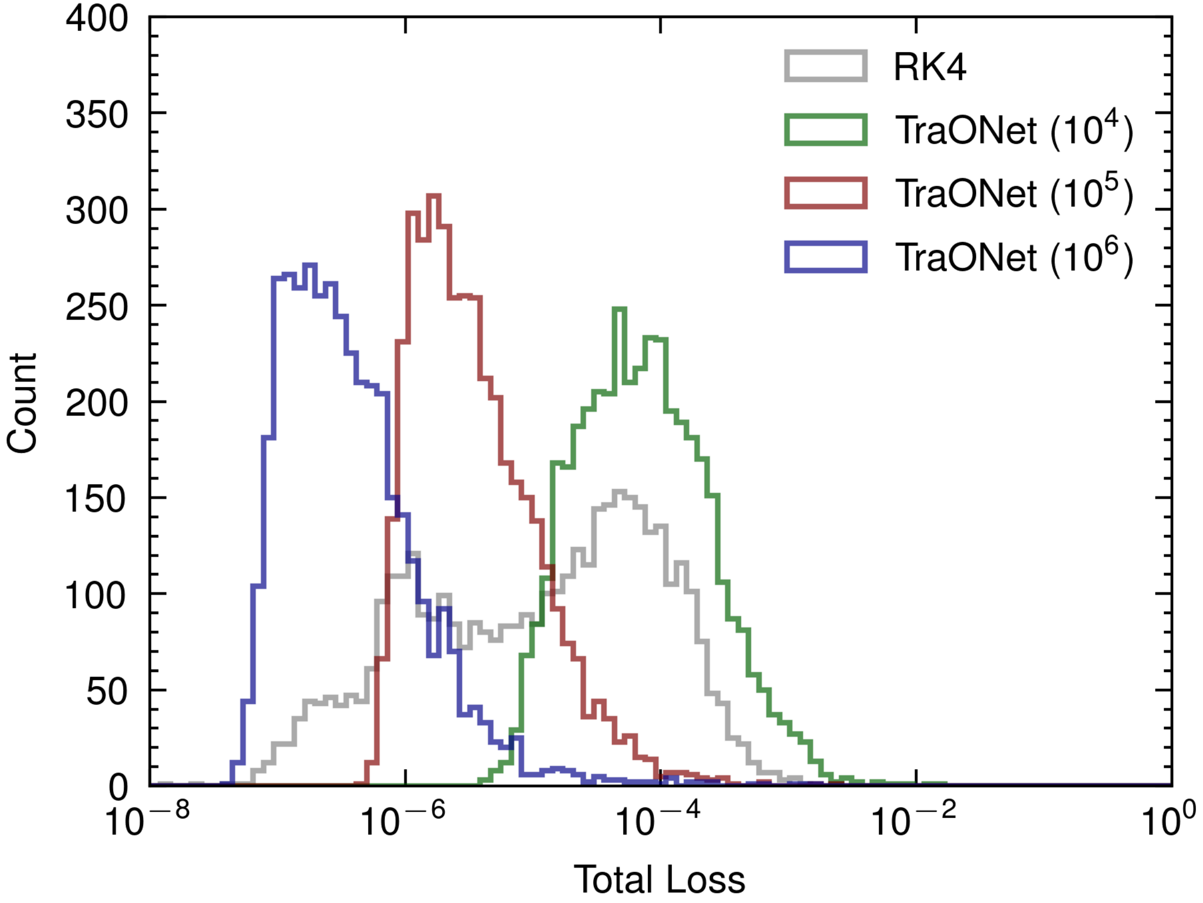


Figure 28: Loss histogram for TraONet trained on different dataset sizes

Thank you

&

Nice to meet you

Supplements

Operator formulation of Hamilton's equation

- Let denote $x(t) = [q(t), p(t)]^T$ then we can rewrite the Hamilton's equation as

$$\begin{cases} \dot{q} = \frac{\partial H}{\partial p} \\ \dot{p} = -\frac{\partial H}{\partial q} \end{cases} \Longrightarrow \dot{x} = J\nabla H \quad \text{where } J = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

- For $H \in C^2(\mathbb{R}^2, \mathbb{R})$ with non divergent $x(t)$, we can find the solution:

$$x(t) = x(0) + \int_0^t J\nabla H[x(\tau)]\mathrm{d}\tau$$

and we can describe it as an operator $G : H(q,p) \mapsto (q(t), p(t))$:

$$G(H)(t) = x(t) = \begin{pmatrix} q(t) \\ p(t) \end{pmatrix}$$

- For simplicity, we assume the kinetic term is $p^2/2$ and consider the operator $\tilde{G}$ as follows.

$$\tilde{G} = G \circ \Phi \quad \text{where} \quad \Phi(V)(q,p) = \frac{p^2}{2} + V(q) \quad \Longrightarrow \quad \tilde{G}[V(q)] = G\left[\frac{p^2}{2} + V(q)\right] = \begin{pmatrix} q(t) \\ p(t) \end{pmatrix}$$