Sampling Neural Networks to Approximate Hamiltonian Functions

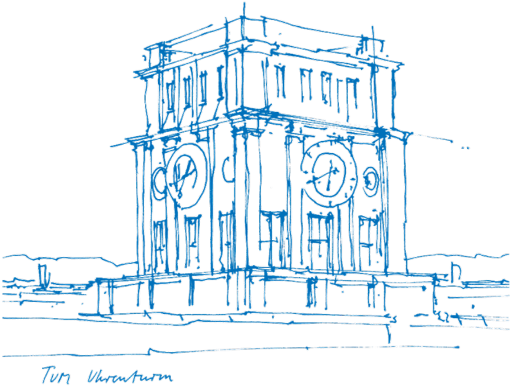
Master's Thesis Talk

Atamert Rahma

Supervisor: Prof. Dr. Felix Dietrich
Advisor: M.Sc. Chinmay Datar

Technical University of Munich
TUM School of Computation, Information and Technology
Chair of Scientific Computing

May 22th, 2024



Outline

Dynamical Systems

"The notion of a dynamical system is the mathematical formalization of the general scientific concept of a deterministic process." [Kuznetsov, 2004].

A dynamical system is a triple $(T, \mathcal{X}, \varphi)$, with

- the time T,
- the state space $\mathcal{X}$,
- the evolution operator $\varphi : T \times \mathcal{X} \to \mathcal{X}$, with
  - $\varphi(0, x) = Id(x) = x$,
  - $\varphi(t + s, x) = \varphi(t, \varphi(s, x)) = (\varphi^t \circ \varphi^s)(x)$ for all $x \in \mathcal{X}$ and $t, s \in T$.

For continuous time dynamical systems ($T = \mathbb{R}_0^+$), when discretized, we write φ_h for the flow with a time step $h > 0$.

Dynamical Systems – Hamiltonian Systems

Hamiltonian Function:

$$\mathcal{H} : \mathcal{X} \to \mathbb{R}.$$

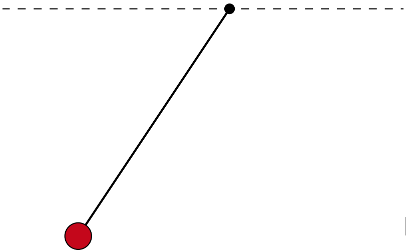
Hamilton's equations [Hamilton, 1834] [Hamilton, 1835]:

$$\dot{q} = \nabla_p \mathcal{H}(q, p)\,, \quad \dot{p} = -\nabla_q \mathcal{H}(q, p)$$

for all $(q, p) \in \mathcal{X}$.

Dynamical Systems – Hamiltonian Systems

Single pendulum system's Hamiltonian: $\mathcal{H}(q, p) = \frac{p^2}{2} + (1 - \cos(q))$.



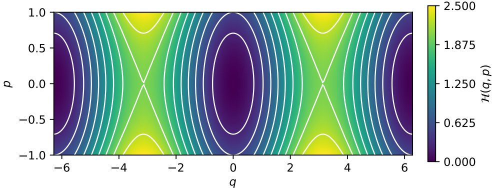


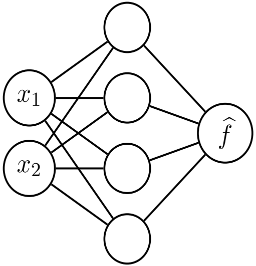
Figure 1 Hamiltonian of the single pendulum.

Outline

Sampling Neural Networks

Traditional network training using gradient-descent has challenges like

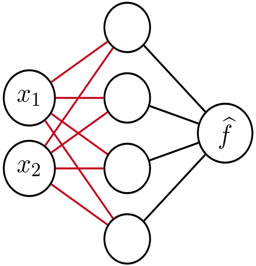
- slow convergence
- learning rate sensitivity
- local minima



Sampling Neural Networks

Traditional network training using gradient-descent has challenges like

- slow convergence
- learning rate sensitivity
- local minima



$\Rightarrow$ **Sample hidden layer parameters!**

- Data-agnostic, e.g., standard normal distribution
- Data-driven using SWIM algorithm [Bolager et al., 2023]

Sampling Neural Networks – SWIM

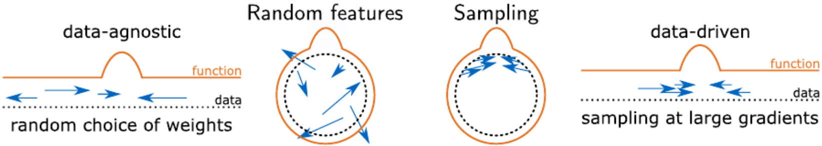


Figure 2 SWIM – Sample Where It Matters: at large gradients, taken from [Bolager et al., 2023].

$\Rightarrow$ Each network parameter is determined by two points from the input space.

The supervised SWIM algorithm defines a pair-sampling probability for each data point pair based on the gradient, **requiring function value information to sample at large gradients**.

Outline

Random Hamiltonian Neural Networks
Approximating the Hamiltonian

Goal: Given observations $\mathcal{D} = \{q_i, p_i, \dot{q}_i, \dot{p}_i\}_{i=1}^{K}$, find an approximation

$$\widehat{\mathcal{H}} = \underset{\widehat{\mathcal{H}}}{\arg\min} \sum_{z \in \mathcal{Z}} \left\|\widehat{\mathcal{H}}(z) - \mathcal{H}(z)\right\|^2,$$

where $\mathcal{Z} \subset \mathcal{X}$.

Approximation using nonlinear functions as

$$\widehat{\mathcal{H}}(x) = \sum_{l \in [L]} w_l \sigma_l(x),$$

where, here, σ represents the output of the last hidden layer of the feed-forward neural network for the given input x.

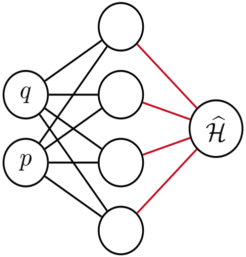
Random Hamiltonian Neural Networks

Approximating the Hamiltonian

Using sampled networks, we only need to fit the last layer's parameters w:

$$\begin{bmatrix} \sigma_1(x_1) & \sigma_2(x_1) & \cdots & \sigma_L(x_1) \\ \sigma_1(x_2) & \sigma_2(x_2) & \cdots & \sigma_L(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_1(x_K) & \sigma_2(x_K) & \cdots & \sigma_L(x_K) \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_L \end{bmatrix} = \begin{bmatrix} \mathcal{H}(x_1) \\ \mathcal{H}(x_2) \\ \vdots \\ \mathcal{H}(x_K) \end{bmatrix}$$

$$\overset{\text{diff.}}{\Longrightarrow} \underbrace{\begin{bmatrix} \nabla\sigma_1(x_1) & \nabla\sigma_2(x_1) & \cdots & \nabla\sigma_L(x_1) \\ \nabla\sigma_1(x_2) & \nabla\sigma_2(x_2) & \cdots & \nabla\sigma_L(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \nabla\sigma_1(x_K) & \nabla\sigma_2(x_K) & \cdots & \nabla\sigma_L(x_K) \end{bmatrix}}_{A} \cdot \underbrace{\begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_L \end{bmatrix}}_{w} = \underbrace{\begin{bmatrix} \nabla\mathcal{H}(x_1) \\ \nabla\mathcal{H}(x_2) \\ \vdots \\ \nabla\mathcal{H}(x_K) \end{bmatrix}}_{u}$$



$\Rightarrow w^* = \underset{w}{\arg\min} \|Aw - u\|^2$ Least Squares Solution: $w^* = (A^T A)^{-1} A^T u$

Random Hamiltonian Neural Networks

We assume we know the function value for a single data point x_0 and write the final linear system with bias (integration constant):

$$\begin{bmatrix} \nabla\sigma_1(x_1) & \nabla\sigma_2(x_1) & \cdots & \nabla\sigma_L(x_1) & 0 \\ \nabla\sigma_1(x_2) & \nabla\sigma_2(x_2) & \cdots & \nabla\sigma_L(x_2) & 0 \\ \vdots & \vdots & \ddots & \vdots & \\ \nabla\sigma_1(x_K) & \nabla\sigma_2(x_K) & \cdots & \nabla\sigma_L(x_K) & 0 \\ \sigma_1(x_0) & \sigma_2(x_0) & \cdots & \sigma_L(x_0) & 1 \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_L \\ b \end{bmatrix} = \begin{bmatrix} \mathcal{J}^{-1}\dot{x}_1 \\ \mathcal{J}^{-1}\dot{x}_2 \\ \vdots \\ \mathcal{J}^{-1}\dot{x}_K \\ \mathcal{H}(x_0) \end{bmatrix},$$

where $\mathcal{J} \cdot \mathcal{H}(x) = \dot{x}$ (Hamilton's equations), where $\mathcal{J} = \begin{bmatrix} 0_d & I_d \\ -I_d & 0_d \end{bmatrix}$ is a $2d \times 2d$ square matrix.

Random Hamiltonian Neural Networks

Given $\mathcal{D} = \{q_i, p_i, \dot{q}_i, \dot{p}_i\}_{i=1}^{K}$ construct a feed-forward neural network.

1. Randomly sample the hidden layer parameters using one of the following.
   - `ELM`: data-agnostic sampling, using **standard normally distributed weights**.
   - `U-SWIM`: data-driven sampling, using **SWIM algorithm with uniformly sampled data pairs**.
   - `A-SWIM`: data-driven sampling, using **SWIM algorithm with initial function value approximations**. $\widehat{\mathcal{H}}_{\texttt{U-SWIM}}(X)$ with the goal of sampling at large gradients.

2. Construct the linear system $Aw = u$.

3. Solve least-squares solution w^* and set the last (linear) layer's weights w.

Outline

Approximations – Single Pendulum with Frequency

$$\mathcal{H}(q,p) = p^2/2 + (1 - \cos(fq))$$

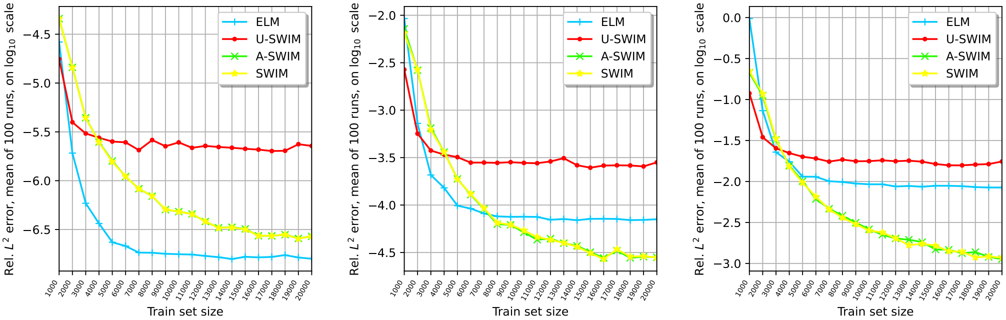


Figure 3 Single pendulum Hamiltonian test errors are plotted. All the settings are trained on domain $[-\pi, \pi] \times [-1, 1]$ with 1500 network width. Frequency (f) is set to 10 at the left, 15 at the center, and 20 at the right plot.

Approximations – Lotka-Volterra

$$\mathcal{H}(q,p) = \beta e^{q} - \alpha q + \delta e^{p} - \gamma p$$

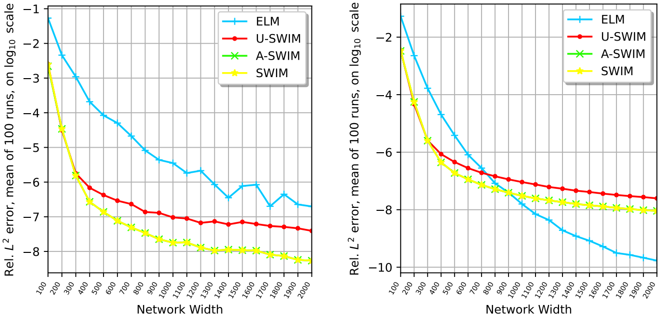
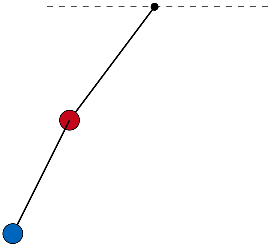


Figure 4 Lotka-Volterra Hamiltonian test errors and gradient train errors are plotted on the left and right, respectively. The function is $(5,5)$ centered and networks are trained on domain $[0,8]^2$ with train set size 10000.

Approximations – Double Pendulum

$$\mathcal{H}(q,p) = \frac{p_1^2 + 2p_2^2 - 2p_1p_2\cos(q_1 - q_2)}{2(1+\sin^2(q_1 - q_2))} - 2\cos(q_1) - \cos(q_2)$$



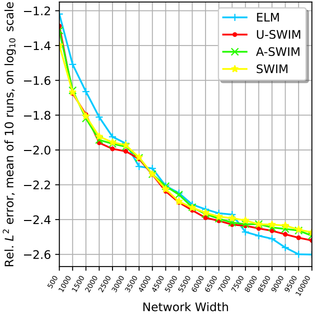


Figure 5 Double pendulum Hamiltonian test errors are plotted on the right. Networks are trained on domain $[-\pi,\pi] \times [-1,1]$ with train set size 20000.

Approximations – Double Pendulum

Integration

$$\mathcal{H}(q,p) = \frac{p_1^2 + 2p_2^2 - 2p_1p_2\cos(q_1 - q_2)}{2(1+\sin^2(q_1 - q_2))} - 2\cos(q_1) - \cos(q_2)$$

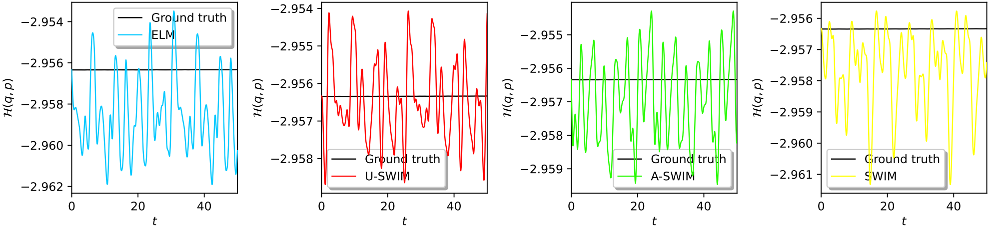


Figure 6 The ground truth Hamiltonian and the trained networks are integrated using symplectic Euler. The initial value is $(0.15, 0.1, -0.05, 0.1)$, integrated until $t = 50$ with $\Delta t = 10^{-4}$.

Approximations – Hénon-Heiles

$$\mathcal{H}(q,p) = \frac{1}{2}(p_1^2 + p_2^2) + \frac{1}{2}(q_1^2 + q_2^2) + \alpha(q_1^2 q_2 - \frac{1}{3}q_2^3)$$

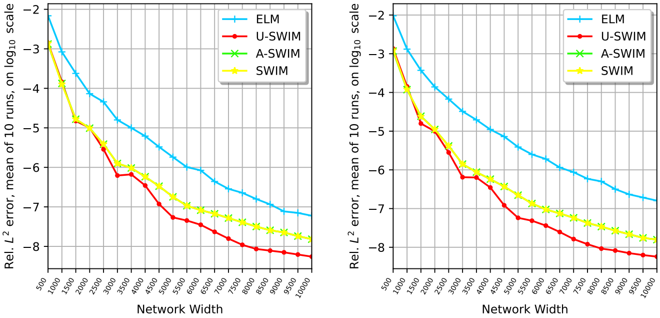


Figure 7 Hénon-Heiles Hamiltonian test errors are plotted with $\alpha = 0$ and $\alpha = 1$ on the left and right plots, respectively. Networks are trained on domain $[-1,1]^2$ with train set size 20000.

Approximations – Poincaré Plots

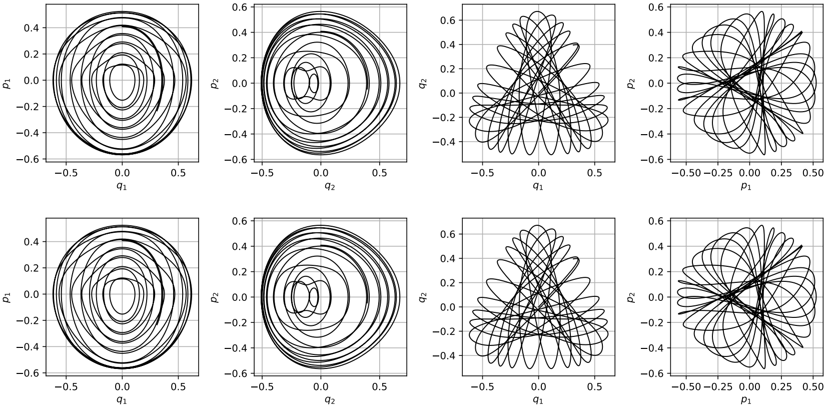


Figure 8 Hénon-Heiles with $\alpha = 0.7$ Poincaré plots are plotted. The top row represents the symplectic Euler using the ground truth Hamiltonian, and the bottom row represents the symplectic Euler using `A-SWIM` approximation. The initial value is $(0, 0, \frac{1}{\sqrt{6}}, \frac{1}{\sqrt{6}})$, integrated until $t = 100$ using $\Delta t = 10^{-3}$.

Approximations – Poincaré Plots

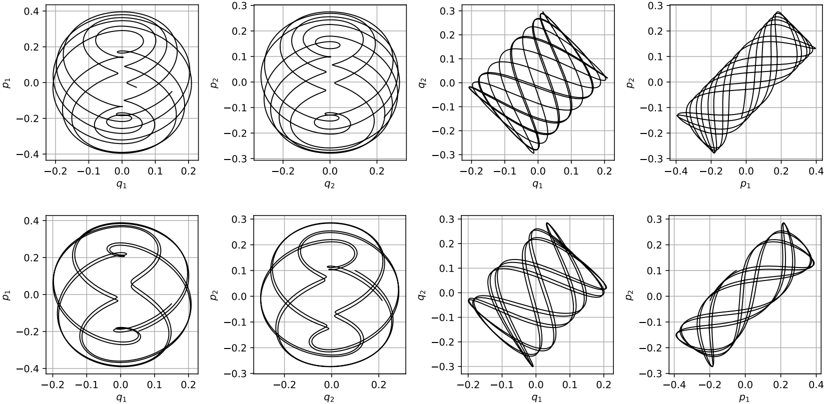


Figure 9 Double pendulum Poincaré plots are plotted. The top row represents the symplectic Euler using the ground truth Hamiltonian, and the bottom row represents the symplectic Euler using `A-SWIM` approximation. The initial value is $(0.15, 0.1, -0.05, 0.1)$, integrated until $t = 50$ with $\Delta t = 10^{-4}$.

Approximations – Limited Data

Given trajectory data $\mathcal{D} = \{q_i, p_i, \varphi_h(q_i, p_i)\}_{i=0}^{K}$, we use finite differences with error correction.

The modified linear system with Symplectic Euler:

$$\begin{bmatrix} \nabla\sigma(\varphi_h(q_1), p_1) & 0 \\ \nabla\sigma(\varphi_h(q_2), p_2) & 0 \\ \vdots & \vdots \\ \nabla\sigma(\varphi_h(q_K), p_K) & 0 \\ \sigma(x_0) & 1 \end{bmatrix} \cdot \begin{bmatrix} w \\ b \end{bmatrix} = \begin{bmatrix} \mathcal{J}^{-1}(\varphi_h(x_1) - x_1)/h \\ \mathcal{J}^{-1}(\varphi_h(x_2) - x_2)/h \\ \vdots \\ \mathcal{J}^{-1}(\varphi_h(x_K) - x_K)/h \\ \mathcal{H}(x_0) \end{bmatrix}.$$

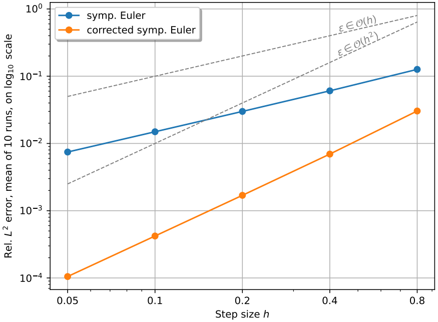


Figure 10 Single pendulum Hamiltonian test errors with and without error correction.

Conclusion & Future Work

- Introduced `Random-HNN` for Hamiltonian approximation given limited data.
- `A-SWIM` used an initial approximation to match the `SWIM`'s performance with limited data.
- `ELM` has better accuracy if the target function is low oscillatory.
- The opposite, `SWIM` has better accuracy if the gradients of the target function are large.
- Error correction can be easily integrated into the linear system.

Potential future work:

- Noisy scenarios can be evaluated.
- Dissipative systems can be studied.
- `Random-HNN` can be configured to approximate the flow map.

Literature

[Kuznetsov, 2004] Y. A. Kuznetsov. Elements of Applied Bifurcation Theory. 2004.

[Hamilton, 1834] W. R. Hamilton. “On a General Method in Dynamics.” In: Philosophical Transactions of the Royal Society 124 (1834), pp. 247–308.

[Hamilton, 1835] W. R. Hamilton. “Second Essay on a General Method in Dynamics.” In: Philosophical Transactions of the Royal Society 125 (1835), pp. 95–144.

[Bolager et al., 2023] E. L. Bolager, I. Burak, C. Datar, Q. Sun, and F. Dietrich. Sampling weights of deep neural networks. 2023.