# TANGO: TIME-REVERSAL LATENT GRAPHODE FOR MULTI-AGENT DYNAMICAL SYSTEMS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Learning complex multi-agent system dynamics from data is crucial across many domains, such as in physical simulations and material modeling. Extended from purely data-driven approaches, existing physics-informed approaches such as Hamiltonian Neural Network strictly follow energy conservation law to introduce inductive bias, making their learning more sample efficiently. However, many real-world systems do not strictly conserve energy, such as spring systems with frictions. Recognizing this, we turn our attention to a broader physical principle: *Time-Reversal Symmetry*, which depicts that the dynamics of a system shall remain invariant when traversed back over time. It still helps to preserve energies for conservative systems and in the meanwhile, serves as a strong inductive bias for non-conservative, reversible systems. To inject such inductive bias, in this paper, we propose a simple-yet-effective self-supervised regularization term as a soft constraint that aligns the forward and backward trajectories predicted by a continuous graph neural network-based ordinary differential equation (GraphODE). It effectively imposes time-reversal symmetry to enable more accurate model predictions across a wider range of dynamical systems under classical mechanics. In addition, we further provide theoretical analysis to show that our regularization essentially minimizes higher-order Taylor expansion terms during the ODE integration steps, which enables our model to be more noise-tolerant and even applicable to irreversible systems. Experimental results on a variety of physical systems demonstrate the effectiveness of our proposed method. Particularly, it achieves an MSE improvement of 11.5 % on a challenging chaotic triple-pendulum systems[1].

## 1 INTRODUCTION

Multi-agent dynamical systems, spanning applications from physical simulations (Battaglia et al., 2016; Kipf et al., 2018; Wang et al., 2020) to robotic control (Li et al., 2022; Gu et al., 2017), are challenging to model due to intricate and dynamic inter-agent interactions. Traditional simulators can be very time-consuming and require domain knowledge of the underlying dynamics, which are often unknown (Sanchez-Gonzalez et al., 2020; Pfaff et al., 2021). Therefore, directly learning a neural simulator from the observational data becomes an attractive alternative. A popular line of research involves using GraphODEs (Huang et al., 2020; Luo et al., 2023; Zang & Wang, 2020), where Graph Neural Networks (GNNs) serve to learn the time integration of the ordinary differential equations(ODEs), for continuous pairwise interactions among agents. Compared with discrete GNN methods (Kipf et al., 2018; Sanchez-Gonzalez et al., 2020), GraphODEs show superior performance in long-range predictions and can handle irregular and partial observations (Jiang et al., 2023).

However, the intricate nature of multi-agent systems often necessitates vast amounts of training data. Vanilla data-driven neural simulators trained on limited datasets tend to be less generalizable, and can violate physical properties of a system such as energy conservation. As depicted in Figure 1 (a.1), the learned energy curve of a baseline model (LG-ODE) (Huang et al., 2020) is prone to explosion, even though the ground-truth energy remains constant. One promising strategy to mitigate this data dependency is to incorporate physical inductive biases (Raissi et al., 2019; Cranmer et al., 2020). Existing works like Hamiltonian- Neural Nets and ODE (Greydanus et al., 2019; Sanchez-Gonzalez et al., 2019) strictly enforce the energy conservation law, leading to more accurate pre-

[1]Code implementation can be found at here.

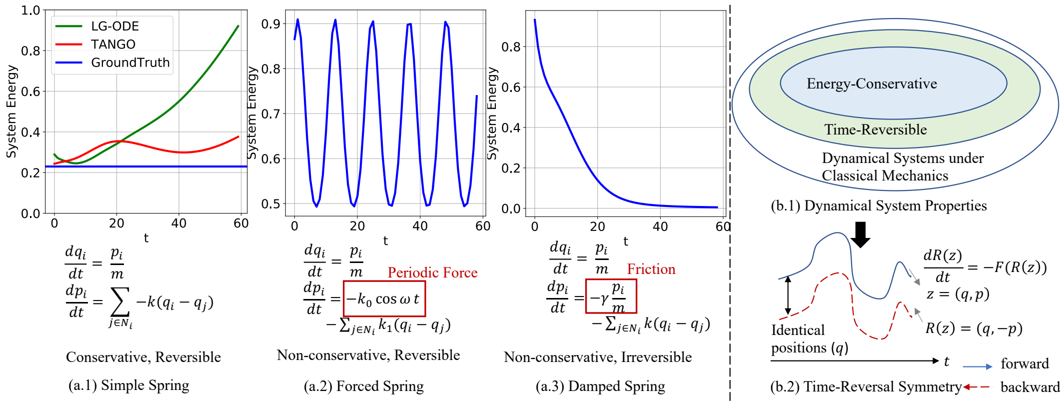


Figure 1: (a) Three $n$-body spring systems characterized by their energy conservation and time reversal properties. $p, q, m$ denote momentum, position and mass, respectively. Proof of energy conservation and time reversal for these systems can be found in Appendix B (b) Classification of classical mechanical systems based on (Tolman, 1938; Lamb & Roberts, 1998)

dictions for some systems under classical mechanics, especially in data-scarce situations. However, not all real-world systems adhere to strict energy conservation, especially those that have interaction with external environments, i.e. non-isolated systems, such as $n$-body spring systems with periodic external forces or frictions shown in Figure 1 (a.2) and (a.3). For these systems, applying strict energy conservation constraint proposed by Greydanus et al. (2019); Sanchez-Gonzalez et al. (2019) could lead to inferior performance. As shown in Figure 1(b.1), for classical and deterministic mechanics such as Newtonian mechanics, energy-conservative systems also obey time-reversal symmetry (Tolman, 1938). On the other hand, we note that the time-reversible systems also include non-conservative systems such as Stokes flow (Pozrikidis, 2001), which also has vital applications in the real world (Kim & Karrila, 2013). Therefore, by ensuring that the system's dynamics is approximately invariant under time reversal, we can enforce neural simulators to generate dynamical systems that are more realistic, paving the way for more efficient and physically coherent dynamical system modeling. In light of these observations, we pivot towards a broader physical principle: *Time-Reversal Symmetry*, which posits that a system's dynamics should remain invariant when time is reversed (Lamb & Roberts, 1998).

To incorporate such time-reversal inductive bias, we propose a simple-yet-effective self-supervised regularization term as a soft constraint that aligns forward and backward trajectories predicted by our model, which has GraphODE as its backbone. We name our model TANGO: **T**ime-Reversal L**a**te**n**t **G**raph **O**DE, which learns the system dynamics in the latent space. This time-reversal loss does not need additional labels beyond ground-truth observations, while effectively imposing time-reversal symmetry to enable more accurate model predictions across a wider range of systems under classical mechanics. Besides its physical implication on benefiting learning *reversible* systems, we also empirically observe that the time-reversal loss in general helps to learn *irreversible* systems. Through theoretical analysis, we prove that from the numerical aspect, the time-reversal loss actually minimizes higher-order Taylor expansion terms during the ODE integration steps, which enables our model to be more noise-tolerable and even applicable to irreversible systems. Therefore, TANGO has the flexibility to be applied to a wide range of dynamical systems without requiring the systems to be strictly energy-conservative or time-reversible. We conducted systematic experiments over four simulated datasets. Experimental results verify the effectiveness of TANGO in learning system dynamics more accurately with less observational data.

The primary contributions of this paper can be summarized as follows:

- We propose TANGO, a GraphODE model that incorporates time-reversal symmetry as a soft constraint and adeptly handles both energy-conservative and non-conservative systems.
- We theoretically explain why the proposed time-reversal symmetry loss could in general help learn more fine-grained and long-context system dynamics from the numerical aspect.
- Our method achieves state-of-the-art results in multi-agent physical simulations. Particularly, it achieves an MSE improvement of 11.5 % on trajectory predictions on a challenging chaotic triple-pendulum system.

# A Theoretical Analysis

## A.1 Proof of Lemma 1

*Proof.* The definition of time-reversal symmetry is given by:

$$R \circ \phi_t = \phi_{-t} \circ R = \phi_t^{-1} \circ R \tag{11}$$

Here, $R$ is an involution operator, which means $R \circ R = \text{id}$.

First, we apply the time evolution operator $\phi_t$ to both sides of Eqn. equation 11:

$$\phi_t \circ R \circ \phi_t = \phi_t \circ \phi_t^{-1} \circ R \tag{12}$$

Simplifying, we obtain:

$$\phi_t \circ R \circ \phi_t = R \tag{13}$$

Next, we apply the involution operator $R$ to both sides of the equation:

$$R \circ \phi_t \circ R \circ \phi_t = R \circ R \tag{14}$$

Since $R \circ R = \text{I}$, we finally arrive at:

$$R \circ \phi_t \circ R \circ \phi_t = \text{I} \tag{15}$$

which means the trajectories can overlap when evolving backward from the final state. □

## A.2 Proof of Theorem 1

Let $\Delta t$ denote the integration step size in an ODE solver and $T$ be the prediction length. The time stamps of the ODE solver are $\{t_j\}_{j=0}^{T}$, where $t_{j+1} - t_j = \Delta t$ for $j = 0, \cdots, T (T > 1)$. Next suppose during the forward evolution, the updates go through states $\boldsymbol{z}^{\text{fwd}}(t_j) = (\boldsymbol{q}^{\text{fwd}}(t_j), \boldsymbol{p}^{\text{fwd}}(t_j))$ for $j = 0, \cdots, T$, where $\boldsymbol{q}^{\text{fwd}}(t_j)$ is position, $\boldsymbol{p}^{\text{fwd}}(t_j)$ is momentum, while during the reverse evolution they go through states $\boldsymbol{z}^{\text{rev}}(t_j) = (\boldsymbol{q}^{\text{rev}}(t_j), \boldsymbol{p}^{\text{rev}}(t_j))$ for $j = 0, \cdots, T$, in reverse order. The ground truth trajectory is $\boldsymbol{z}^{\text{gt}}(t_j) = (\boldsymbol{q}^{\text{gt}}(t_j), \boldsymbol{p}^{\text{gt}}(t_j))$ for $j = 0, \cdots, T$.

For the sake of brevity in the ensuing proof, we denote $\boldsymbol{z}^{\text{gt}}(t_j)$ by $\boldsymbol{z}_j^{\text{gt}}$, $\boldsymbol{z}^{\text{fwd}}(t_j)$ by $\boldsymbol{z}_j^{\text{fwd}}$ and $\boldsymbol{z}^{\text{rev}}(t_j)$ by $\boldsymbol{z}_j^{\text{rev}}$, and we will use Mathematical Induction to prove the theorem.

### A.2.1 Reconstruction Loss ($\mathcal{L}_{pred}$) Analysis.

First, we bound the forward loss $\sum_{j=0}^{T} \|\boldsymbol{z}_j^{\text{fwd}} - \boldsymbol{z}_j^{\text{gt}}\|_2^2$. Since our method models the momentum and position of the system, we can write the following Taylor expansion of the forward process, where for any $0 \le j < T$:

$$\begin{cases} \boldsymbol{q}_{j+1}^{\text{fwd}} = \boldsymbol{q}_j^{\text{fwd}} + (\boldsymbol{p}_j^{\text{fwd}}/m)\Delta t + (\dot{\boldsymbol{p}}_j^{\text{fwd}}/2m)\Delta t^2 + \mathcal{O}(\Delta t^3), & \text{(16a)} \\ \boldsymbol{p}_{j+1}^{\text{fwd}} = \boldsymbol{p}_j^{\text{fwd}} + \dot{\boldsymbol{p}}_j^{\text{fwd}}\Delta t + \mathcal{O}(\Delta t^2), & \text{(16b)} \\ \dot{\boldsymbol{p}}_{j+1}^{\text{fwd}} = \dot{\boldsymbol{p}}_j^{\text{fwd}} + \mathcal{O}(\Delta t), & \text{(16c)} \end{cases}$$

and for the ground truth process, we also have from Taylor expansion that

$$\begin{cases} \boldsymbol{q}_{j+1}^{\text{gt}} = \boldsymbol{q}_j^{\text{gt}} + (\boldsymbol{p}_j^{\text{gt}}/m)\Delta t + (\dot{\boldsymbol{p}}_j^{\text{gt}}/2m)\Delta t^2 + \mathcal{O}(\Delta t^3), & \text{(17a)} \\ \boldsymbol{p}_{j+1}^{\text{gt}} = \boldsymbol{p}_j^{\text{gt}} + \dot{\boldsymbol{p}}_j^{\text{gt}}\Delta t + \mathcal{O}(\Delta t^2), & \text{(17b)} \\ \dot{\boldsymbol{p}}_{j+1}^{\text{gt}} = \dot{\boldsymbol{p}}_j^{\text{gt}} + \mathcal{O}(\Delta t). & \text{(17c)} \end{cases}$$

With these, we aim to prove that for any $k = 0, 1, \cdots, T$, the following hold :

$$\begin{cases} \|\boldsymbol{q}_k^{\text{fwd}} - \boldsymbol{q}_k^{\text{gt}}\|_2 \le C_2^{\text{fwd}} k^2 \Delta t^2, & \text{(18a)} \\ \|\boldsymbol{p}_k^{\text{fwd}} - \boldsymbol{p}_k^{\text{gt}}\|_2 \le C_1^{\text{fwd}} k \Delta t, & \text{(18b)} \end{cases}$$

where $C_1^{\text{fwd}}$ and $C_2^{\text{fwd}}$ are constants.

**Base Case** $k = 0$**:** Based on the initialization rules, it is obvious that $\left\|\boldsymbol{q}_0^{\text{fwd}} - \boldsymbol{q}_0^{\text{gt}}\right\|_2 = 0$ and $\left\|\boldsymbol{p}_0^{\text{fwd}} - \boldsymbol{p}_0^{\text{gt}}\right\|_2 = 0$, thus (18a) and (18b) both hold for $k = 0$.

**Inductive Hypothesis:** Assume (18a) and (18b) hold for $k = j$, which means:

$$\begin{cases} \|\boldsymbol{q}_j^{\text{fwd}} - \boldsymbol{q}_j^{\text{gt}}\|_2 \le C_2^{\text{fwd}} j^2 \Delta t^2, & (19a) \\ \|\boldsymbol{p}_j^{\text{fwd}} - \boldsymbol{p}_j^{\text{gt}}\|_2 \le C_1^{\text{fwd}} j \Delta t, & (19b) \end{cases}$$

**Inductive Proof:** We need to prove (18a) and (18b) hold for $k = j + 1$.

First, using (16c) and (17c), we have

$$\left\|\dot{\boldsymbol{p}}_{j+1}^{\text{fwd}} - \dot{\boldsymbol{p}}_{j+1}^{\text{gt}}\right\|_2 = \left\|\dot{\boldsymbol{p}}_j^{\text{fwd}} - \dot{\boldsymbol{p}}_j^{\text{gt}}\right\|_2 + \mathcal{O}(\Delta t) = \left\|\dot{\boldsymbol{p}}_0^{\text{fwd}} - \dot{\boldsymbol{p}}_0^{\text{gt}}\right\|_2 + \mathcal{O}\big((j+1)\Delta t\big) = \mathcal{O}(1), \quad (20)$$

where we iterate through $j, j-1, \cdots, 0$ in the second equality. Then using (17b) and (16b), we get for $j + 1$ that

$$\begin{aligned} \left\|\boldsymbol{p}_{j+1}^{\text{fwd}} - \boldsymbol{p}_{j+1}^{\text{gt}}\right\|_2 &= \left\|\big(\boldsymbol{p}_j^{\text{fwd}} + \dot{\boldsymbol{p}}_j^{\text{fwd}}\Delta t\big) - \big(\boldsymbol{p}_j^{\text{gt}} + \dot{\boldsymbol{p}}_j^{\text{gt}}\Delta t\big) + \mathcal{O}(\Delta t^2)\right\|_2 \\ &\le \left\|\boldsymbol{p}_j^{\text{fwd}} - \boldsymbol{p}_j^{\text{gt}}\right\|_2 + \left\|\dot{\boldsymbol{p}}_j^{\text{fwd}} - \dot{\boldsymbol{p}}_j^{\text{gt}}\right\|_2 \Delta t + \mathcal{O}(\Delta t^2) \\ &\le \big[C_1^{\text{fwd}} j + \mathcal{O}(1)\big]\Delta t, \end{aligned}$$

where the first inequality uses the triangle inequality, and in the second inequality we use (19b) as well as (20). We can see there exists $C_1^{\text{fwd}}$ such that the final expression above is upper bounded by $C_1^{\text{fwd}}(j + 1)\Delta t$, with which the claim holds for $j + 1$.

Next for (18a), using (17a) and (16a), we get for any $j$ that

$$\begin{aligned} \left\|\boldsymbol{q}_{j+1}^{\text{fwd}} - \boldsymbol{q}_{j+1}^{\text{gt}}\right\|_2 &= \Big\|\big(\boldsymbol{q}_j^{\text{fwd}} + (\boldsymbol{p}_j^{\text{fwd}}/m)\Delta t + (\dot{\boldsymbol{p}}_j^{\text{fwd}}/2m)\Delta t^2\big) \\ &\qquad - \big(\boldsymbol{q}_j^{\text{gt}} + (\boldsymbol{p}_j^{\text{gt}}/m)\Delta t + (\dot{\boldsymbol{p}}_j^{\text{gt}}/2m)\Delta t^2\big) + \mathcal{O}(\Delta t^3)\Big\|_2 \\ &\le \left\|\boldsymbol{q}_j^{\text{fwd}} - \boldsymbol{q}_j^{\text{gt}}\right\|_2 + \frac{1}{m}\left\|\boldsymbol{p}_j^{\text{fwd}} - \boldsymbol{p}_j^{\text{gt}}\right\|_2 \Delta t + \frac{1}{2m}\left\|\dot{\boldsymbol{p}}_j^{\text{fwd}} - \dot{\boldsymbol{p}}_j^{\text{gt}}\right\|_2 \Delta t^2 + \mathcal{O}(\Delta t^3) \\ &\le \left[C_2^{\text{fwd}} j^2 + \frac{C_1^{\text{fwd}}}{m} j + \mathcal{O}(1)\right]\Delta t^2, \end{aligned}$$

where the first inequality uses the triangle inequality, and in the second inequality we use (19a) and (19b) as well as (20). Thus with an appropriate $C_2^{\text{fwd}}$, we have the final expression above is upper bounded by $C_2^{\text{fwd}}(j + 1)^2\Delta t^2$, and so the claim holds for $j + 1$.

Since both the base case and the inductive step have been proven, by the principle of mathematical induction, (18a) and (18b) holds for all $k = 0, 1, \cdots, T$.

With this, we finish the forward proof by plugging (18a) and (18b) into the loss function:

$$\begin{aligned} \sum_{j=0}^{T} \|\boldsymbol{z}_j^{\text{fwd}} - \boldsymbol{z}_j^{\text{gt}}\|_2^2 &= \sum_{j=0}^{T} \|\boldsymbol{p}_j^{\text{fwd}} - \boldsymbol{p}_j^{\text{gt}}\|_2^2 + \sum_{j=0}^{T} \|\boldsymbol{q}_j^{\text{fwd}} - \boldsymbol{q}_j^{\text{gt}}\|_2^2 \\ &\le \big(C_1^{\text{fwd}}\big)^2 \sum_{j=0}^{T} j^2\Delta t^2 + \big(C_2^{\text{fwd}}\big)^2 \sum_{j=0}^{T} j^4\Delta t^4 \\ &= \mathcal{O}(T^3\Delta t^2). \end{aligned}$$

### A.2.2 Reversal Loss ($\mathcal{L}_{reverse}$) Analysis.

Next we analyze the reversal loss $\sum_{j=0}^{T} \|R(\boldsymbol{z}_j^{\text{rev}}) - \boldsymbol{z}_j^{\text{fwd}}\|_2^2$. For this, we need to refine the Taylor expansion residual terms for a more in-depth analysis.

First reconsider the forward process. Since the process is generated from the learned network, we may assume that for some constants $c_1$, $c_2$, and $c_3$, the states satisfy the following for any $0 \le j < T$:

$$\begin{cases} \boldsymbol{q}_j^{\text{fwd}} = \boldsymbol{q}_{j+1}^{\text{fwd}} - (\boldsymbol{p}_{j+1}^{\text{fwd}}/m)\Delta t + (\dot{\boldsymbol{p}}_{j+1}^{\text{fwd}}/2m)\Delta t^2 + \mathbf{rem}_j^{\text{fwd},3}, & (21a) \\ \boldsymbol{p}_j^{\text{fwd}} = \boldsymbol{p}_{j+1}^{\text{fwd}} - \dot{\boldsymbol{p}}_{j+1}^{\text{fwd}}\Delta t + \mathbf{rem}_j^{\text{fwd},2}, & (21b) \\ \dot{\boldsymbol{p}}_j^{\text{fwd}} = \dot{\boldsymbol{p}}_{j+1}^{\text{fwd}} + \mathbf{rem}_j^{\text{fwd},1}, & (21c) \end{cases}$$

where the remaining terms $\left\|\mathbf{rem}_j^{\text{fwd},i}\right\|_2 \leq c_i \Delta t^i$ for $i = 1, 2, 3$. Similarly, we have approximate Taylor expansions for the reverse process:

$$\begin{cases} \boldsymbol{q}_j^{\text{rev}} = \boldsymbol{q}_{j+1}^{\text{rev}} + (\boldsymbol{p}_{j+1}^{\text{rev}}/m)\Delta t + (\dot{\boldsymbol{p}}_{j+1}^{\text{rev}}/2m)\Delta t^2 + \mathbf{rem}_j^{\text{rev},3}, & \text{(22a)} \\ \boldsymbol{p}_j^{\text{rev}} = \boldsymbol{p}_{j+1}^{\text{rev}} + \dot{\boldsymbol{p}}_{j+1}^{\text{rev}}\Delta t + \mathbf{rem}_j^{\text{rev},2}, & \text{(22b)} \\ \dot{\boldsymbol{p}}_j^{\text{rev}} = \dot{\boldsymbol{p}}_{j+1}^{\text{rev}} + \mathbf{rem}_j^{\text{rev},1}, & \text{(22c)} \end{cases}$$

where $\left\|\mathbf{rem}_j^{\text{rev},i}\right\|_2 \leq c_i \Delta t^i$ for $i = 1, 2, 3$.

We will prove via induction that for $k = T, T-1, \cdots, 0$,

$$\begin{cases} \|R(\boldsymbol{q}_k^{\text{rev}}) - \boldsymbol{q}_k^{\text{fwd}}\|_2 \leq C_3^{\text{rev}}(T-k)^3\Delta t^3, & \text{(23a)} \\ \|R(\boldsymbol{p}_k^{\text{rev}}) - \boldsymbol{p}_k^{\text{fwd}}\|_2 \leq C_2^{\text{rev}}(T-k)^2\Delta t^2, & \text{(23b)} \\ \|R(\dot{\boldsymbol{p}}_k^{\text{rev}}) - \dot{\boldsymbol{p}}_k^{\text{fwd}}\|_2 \leq C_1^{\text{rev}}(T-k)\Delta t, & \text{(23c)} \end{cases}$$

where $C_1^{\text{rev}}$, $C_2^{\text{rev}}$ and $C_3^{\text{rev}}$ are constants.

The entire proof process is analogous to the previous analysis of Reconstruction Loss.

**Base Case $k = T$:** Since the reverse process is initialized by the forward process variables at $k = T$, it is obvious that $\left\|\boldsymbol{q}_T^{\text{fwd}} - \boldsymbol{q}_T^{ev}\right\|_2 = \left\|\boldsymbol{p}_T^{\text{fwd}} - \boldsymbol{p}_T^{\text{rev}}\right\|_2 = \left\|\dot{\boldsymbol{p}}_T^{\text{fwd}} - \dot{\boldsymbol{p}}_T^{\text{rev}}\right\|_2 = 0$. Thus (23a), (23b) and (23c) all hold for $k = 0$.

**Inductive Hypothesis:** Assume the inequalities (23b), (23a) and (23c) hold for $k = j + 1$, which means:

$$\begin{cases} \|R(\boldsymbol{q}_{j+1}^{\text{rev}}) - \boldsymbol{q}_{j+1}^{\text{fwd}}\|_2 \leq C_3^{\text{rev}}(T-(j+1))^3\Delta t^3, & \text{(24a)} \\ \|R(\boldsymbol{p}_{j+1}^{\text{rev}}) - \boldsymbol{p}_{j+1}^{\text{fwd}}\|_2 \leq C_2^{\text{rev}}(T-(j+1))^2\Delta t^2, & \text{(24b)} \\ \|R(\dot{\boldsymbol{p}}_{j+1}^{\text{rev}}) - \dot{\boldsymbol{p}}_{j+1}^{\text{fwd}}\|_2 \leq C_1^{\text{rev}}(T-(j+1))\Delta t, & \text{(24c)} \end{cases}$$

**Inductive Proof**: We need to prove (23b) (23a) and (23c) holds for $k = j$.

First, for (23c), using (21c) and (22c), we get for any $j$ that

$$\begin{aligned} \left\|R(\dot{\boldsymbol{p}}_j^{\text{rev}}) - \dot{\boldsymbol{p}}_j^{\text{fwd}}\right\|_2 &= \left\|(\dot{\boldsymbol{p}}_{j+1}^{\text{rev}} + \mathbf{rem}_j^{\text{rev},1}) - (\dot{\boldsymbol{p}}_{j+1}^{\text{fwd}} + \mathbf{rem}_j^{\text{fwd},1})\right\|_2 \\ &\leq \left\|R(\dot{\boldsymbol{p}}_{j+1}^{\text{rev}}) - \dot{\boldsymbol{p}}_{j+1}^{\text{fwd}}\right\|_2 + \|\mathbf{rem}_j^{\text{rev},1}\|_2 + \|\mathbf{rem}_j^{\text{fwd},1}\|_2 \\ &\leq C_1^{\text{rev}}(T-j-1)\Delta t + 2c_1\Delta t, \end{aligned}$$

where the first inequality uses the triangle inequality, and the second inequality plugs in (24c). Thus taking $C_1^{\text{rev}} = 2c_1$, the above is upped bounded by $C_1^{\text{rev}}(T-j)\Delta t$, and (23b) holds for $j$.

Second, for (24b), using (21b) and (22b), we get

$$\begin{aligned} \left\|R(\boldsymbol{p}_j^{\text{rev}}) - \boldsymbol{p}_j^{\text{fwd}}\right\|_2 &= \left\| -\left(\boldsymbol{p}_{j+1}^{\text{rev}} + \dot{\boldsymbol{p}}_{j+1}^{\text{rev}}\Delta t + \mathbf{rem}_j^{\text{rev},2}\right) - \left(\boldsymbol{p}_{j+1}^{\text{fwd}} - \dot{\boldsymbol{p}}_{j+1}^{\text{fwd}}\Delta t + \mathbf{rem}_j^{\text{fwd},2}\right)\right\|_2 \\ &\leq \left\|R(\boldsymbol{p}_{j+1}^{\text{rev}}) - \boldsymbol{p}_{j+1}^{\text{fwd}}\right\|_2 + \left\|R(\dot{\boldsymbol{p}}_{j+1}^{\text{rev}}) - \dot{\boldsymbol{p}}_{j+1}^{\text{fwd}}\right\|_2\Delta t \\ &\quad + \|\mathbf{rem}_j^{\text{rev},2}\|_2 + \|\mathbf{rem}_j^{\text{fwd},2}\|_2 \\ &\leq \left[C_2^{\text{rev}}(T-j-1)^2 + C_1^{\text{rev}}(T-j-1) + 2c_2\right]\Delta t^2, \end{aligned}$$

where the first inequality uses the triangle inequality, and in the second inequality we use (24a) and (24b). Thus taking $C_2^{\text{rev}} = \max\{C_1^{\text{rev}}/2, 2c_2\}$, we have the final expression above is upper bounded by $C_2^{\text{rev}}(T-j)^2\Delta t^2$, and so the claim holds for $j$.

Finally, for (24a), we use (21a) and (22a) to get

$$\begin{aligned} \left\|R(\boldsymbol{q}_j^{\text{rev}}) - \boldsymbol{q}_j^{\text{fwd}}\right\|_2 &= \left\|\left(\boldsymbol{q}_{j+1}^{\text{rev}} + (\boldsymbol{p}_{j+1}^{\text{rev}}/m)\Delta t + (\dot{\boldsymbol{p}}_{j+1}^{\text{rev}}/2m)\Delta t^2 + \mathbf{rem}_j^{\text{rev},3}\right)\right. \\ &\qquad \left. - \left(\boldsymbol{q}_{j+1}^{\text{fwd}} - (\boldsymbol{p}_{j+1}^{\text{fwd}}/m)\Delta t + (\dot{\boldsymbol{p}}_{j+1}^{\text{fwd}}/2m)\Delta t^2 + \mathbf{rem}_j^{\text{fwd},3}\right)\right\|_2 \\ &\leq \left\|R(\boldsymbol{q}_{j+1}^{\text{rev}}) - \boldsymbol{q}_{j+1}^{\text{fwd}}\right\|_2 + \frac{1}{m}\left\|R(\boldsymbol{p}_{j+1}^{\text{rev}}) - \boldsymbol{p}_{j+1}^{\text{fwd}}\right\|_2\Delta t \\ &\quad + \frac{1}{2m}\left\|R(\dot{\boldsymbol{p}}_{j+1}^{\text{rev}}) - \dot{\boldsymbol{p}}_{j+1}^{\text{fwd}}\right\|_2\Delta t^2 + \|\mathbf{rem}_j^{\text{rev},3}\|_2 + \|\mathbf{rem}_j^{\text{fwd},3}\|_2 \\ &\leq \left[C_3^{\text{rev}}(T-j-1)^3 + \frac{C_2^{\text{rev}}}{m}(T-j-1)^2 + \frac{C_1^{\text{rev}}}{2m}(T-j-1) + 2c_3\right]\Delta t^3, \end{aligned}$$

where the first inequality uses the triangle inequality, and in the second inequality we use (24a), (24b) and (24c). Thus taking $C_3^{\text{rev}} = \max\{C_2^{\text{rev}}/3m, C_1^{\text{rev}}/6m, 2c_3\}$, we have the final expression above is upper bounded by $C_3^{\text{rev}}(T-j)^3\Delta t^3$, and so the claim holds for $j$.

Since both the base case and the inductive step have been proven, by the principle of mathematical induction, (23b), (23a) and (23c) hold for all $k = T, T-1, \cdots, 0$.

With this we finish the proof by plugging (23b) and (23a) into the loss function:

$$\begin{aligned}\sum_{j=0}^{T} \|R(\boldsymbol{z}_j^{\text{rev}}) - \boldsymbol{z}_j^{\text{fwd}}\|_2^2 &= \sum_{j=0}^{T} \|R(\boldsymbol{p}_j^{\text{rev}}) - \boldsymbol{p}_j^{\text{fwd}}\|_2^2 + \sum_{j=0}^{T} \|R(\boldsymbol{q}_j^{\text{rev}}) - \boldsymbol{q}_j^{\text{fwd}}\|_2^2 \\ &\leq \left(C_2^{\text{rev}}\right)^2 \sum_{j=0}^{T} (T-j)^4 \Delta t^4 + \left(C_3^{\text{rev}}\right)^2 \sum_{j=0}^{T} (T-j)^6 \Delta t^6 \\ &= \mathcal{O}(T^5 \Delta t^4).\end{aligned}$$

### A.3 Analysis on Implementations of Reversal Loss

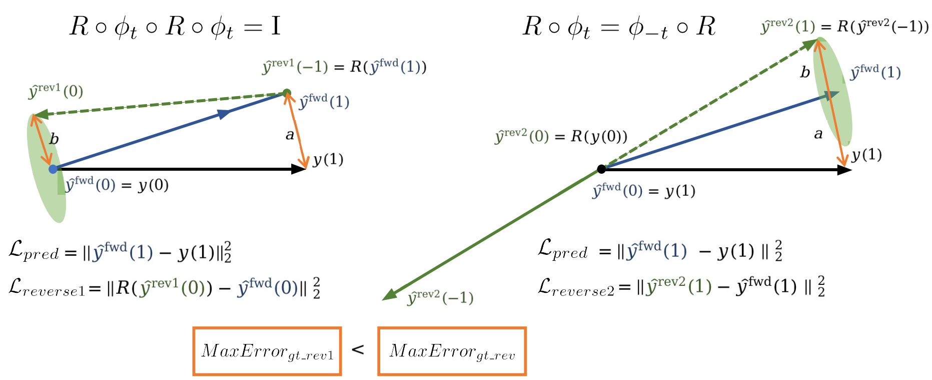


Figure 7: Comparison between two reversal loss implementation

Our time-reversal loss implementation builts upon Lemma1 where the backward trajectory originates from the last state of the forward trajectory. One could also implement the reversal loss based on Eqn 5 which is adopted in TRS-ODEN (Huh et al., 2020). We illustrated the comparison between the two implementation in Figure 7.

Here, we provide the following Lemma to show their difference:

**Lemma 2.** *Comparing our reversal loss implementation against the implementation following Eqn.(5), when the reconstruction loss defined in Eqn.(7) and the time-reversal loss defined in Eqn. (9) both have the same value between the two methods, the maximum error between the reversal and ground truth trajectory, i.e.* $MaxError_{gt\_rev} = \max_{j \in [T]} \|\boldsymbol{y}(j) - \hat{\boldsymbol{y}}^{\text{rev}}(T-j)\|_2$ *made by our method is smaller than that in the second method.*

Lemma 2 suggests that our implementation of time-reversal symmetry is numerically better than the implementation used in (Huh et al., 2020). We give the detailed proof below.

We expect an ideal model to align both the predicted forward and reverse trajectories with the ground truth. As shown in Figure 7, we integrate one step from the initial state $\hat{\boldsymbol{y}}^{\text{fwd}}(0)$ (which is the same as $\boldsymbol{y}(0)$) and reach the state $\hat{\boldsymbol{y}}^{\text{fwd}}(1)$.

The first reverse loss implementation (ours) follows lemma (1) as $R \circ \Phi_t \circ R \circ \Phi_t = id$, which means when we evolve forward and reach the state $\hat{\boldsymbol{y}}^{\text{fwd}}(1)$ we reverse it into $\hat{\boldsymbol{y}}^{\text{rev1}}(-1)$ and go back to reach $\hat{\boldsymbol{y}}^{\text{rev1}}(0)$, then reverse it to get $R(\hat{\boldsymbol{y}}^{\text{rev1}}(0))$, which ideally should be the same as $\hat{\boldsymbol{y}}^{\text{fwd}}(0)$.

The second reverse loss implementation follows Eqn.(5) as $R \circ \Phi_t = \Phi_{-t} \circ R$, which means we first reverse the initial state as $\hat{\boldsymbol{y}}^{\text{rev2}}(0) = R(\boldsymbol{y}(0))$, then evolve the reverse trajectory in the opposite direction to reach $\hat{\boldsymbol{y}}^{\text{rev2}}(-1)$, and then perform a time-symmetric operation to reach $\hat{\boldsymbol{y}}^{\text{rev2}}(1)$, aligning it with the forward trajectory.

We assume the two reconstruction losses $\mathcal{L}_{pred} = \|\hat{\boldsymbol{y}}^{\text{fwd}}(1) - \boldsymbol{y}(1)\|_2^2 := a$ are the same. For the time-reversal losses:

$$\mathcal{L}_{reverse1} = \|R(\hat{\boldsymbol{y}}^{\text{rev1}}(0)) - \hat{\boldsymbol{y}}^{\text{fwd}}(0)\|_2^2 + \|R(\hat{\boldsymbol{y}}^{\text{rev1}}(-1)) - \hat{\boldsymbol{y}}^{\text{fwd}}(1)\|_2^2 = \|R(\hat{\boldsymbol{y}}^{\text{rev1}}(0)) - \hat{\boldsymbol{y}}^{\text{fwd}}(0)\|_2^2 := b,$$
$$\mathcal{L}_{reverse2} = \|\hat{\boldsymbol{y}}^{\text{rev2}}(0) - \hat{\boldsymbol{y}}^{\text{fwd}}(0)\|_2^2 + \|\hat{\boldsymbol{y}}^{\text{rev2}}(1) - \hat{\boldsymbol{y}}^{\text{fwd}}(1)\|_2^2 = \|\hat{\boldsymbol{y}}^{\text{rev2}}(1) - \hat{\boldsymbol{y}}^{\text{fwd}}(1)\|_2^2 := b,$$

we also assume they have reached the same value $b$.

As shown in Figure.7 where we illustrate the worse case scenario, we can see that:

$$MaxError_{gt\_rev1} = \max\left\{\left\|R(\hat{\boldsymbol{y}}^{\text{rev1}}(0)) - \boldsymbol{y}(0)\right\|_2, \left\|R(\hat{\boldsymbol{y}}^{\text{rev1}}(-1)) - \boldsymbol{y}(1)\right\|_2\right\} = max\{a, b\},$$
$$MaxError_{gt\_rev2} = \left\|\hat{\boldsymbol{y}}^{\text{rev2}}(1) - \hat{\boldsymbol{y}}^{\text{fwd}}(1)\right\|_2 + \left\|\hat{\boldsymbol{y}}^{\text{fwd}}(1) - \boldsymbol{y}(1)\right\|_2 = a + b,$$

This means our model achieves a smaller error of the maximum distance between the reversal and ground truth trajectory.

## B Example of varying dynamical systems

We illustrate the energy conservation and time reversal of the three n-body spring systems in Figure 1(a). We use the Hamiltonian formalism of systems under classical mechanics to describe their dynamics and verify their energy conservation and time-reversibility characteristics.

The scalar function that describes a systems motion is called the Hamiltonian, $\mathcal{H}$, and is typically equal to the total energy of the system, that is, the potential energy plus the kinetic energy (North, 2021). It describes the phase space equations of motion by following two first-order ODEs called Hamilton's equations:

$$\frac{d\mathbf{q}}{dt} = \frac{\partial \mathcal{H}(\mathbf{q}, \mathbf{p})}{\partial \mathbf{p}}, \frac{d\mathbf{p}}{dt} = -\frac{\partial \mathcal{H}(\mathbf{q}, \mathbf{p})}{\partial \mathbf{q}}, \tag{25}$$

where $\mathbf{q} \in \mathbb{R}^n$, $\mathbf{p} \in \mathbb{R}^n$, and $\mathcal{H} : \mathbb{R}^{2n} \mapsto \mathbb{R}$ are positions, momenta, and Hamiltonian of the system.

Under this formalism, energy conservative is defined by $d\mathcal{H}/dt = 0$, and the time-reversal symmetry is defined by $\mathcal{H}(q, p, t) = \mathcal{H}(q, -p, -t)$ (Lamb & Roberts, 1998).

### B.1 Conservative and reversible systems.

A simple example is the isolated n-body spring system, which can be described by :

$$\begin{aligned} \frac{d\mathbf{q_i}}{dt} &= \frac{\mathbf{p_i}}{m} \\ \frac{d\mathbf{p_i}}{dt} &= \sum_{j \in N_i} -k(\mathbf{q_i} - \mathbf{q_j}), \end{aligned} \tag{26}$$

where $\mathbf{q} = (\mathbf{q_1}, \mathbf{q_2}, \cdots, \mathbf{q_N})$ is a set of positions of each object , $\mathbf{p} = (\mathbf{p_1}, \mathbf{p_2}, \cdots, \mathbf{p_N})$ is a set of momenta of each object, $m_i$ is mass of each object, $k$ is spring constant.

The Hamilton's equations are:

$$\begin{aligned} \frac{\partial \mathcal{H}(\mathbf{q}, \mathbf{p})}{\partial \mathbf{p_i}} &= \frac{d\mathbf{q_i}}{dt} = \frac{\mathbf{p_i}}{m} \\ \frac{\partial \mathcal{H}(\mathbf{q}, \mathbf{p})}{\partial \mathbf{q_i}} &= -\frac{d\mathbf{p_i}}{dt} = \sum_{j \in N_i} k(\mathbf{q_i} - \mathbf{q_j}), \end{aligned} \tag{27}$$

Hence, we can obtain the Hamiltonian through the integration of the above equation.

$$\mathcal{H}(\mathbf{q}, \mathbf{p}) = \sum_{i=1}^{N} \frac{\mathbf{p_i}^2}{2m_i} + \frac{1}{2} \sum_{i=1}^{N} \sum_{j \in N_i}^{N} \frac{1}{2} k(\mathbf{q_i} - \mathbf{q_j})^2, \tag{28}$$

**Verify the systems' energy conservation**

$$\frac{d\mathcal{H}(\mathbf{q},\mathbf{p})}{dt} = \frac{1}{dt}(\sum_{i=1}^{N}\frac{\mathbf{p_i}^2}{2m_i}) + \frac{1}{dt}\big(\frac{1}{2}\sum_{i=1}^{N}\sum_{j\in N_i}^{N}\frac{1}{2}k(\mathbf{q_i}-\mathbf{q_j})^2\big) = 0, \tag{29}$$

So it is conservative.

**Verify the systems' time-reversal symmetry** We do the transformation $R : (\mathbf{q},\mathbf{p},t) \mapsto (\mathbf{q},-\mathbf{p},-t)$.

$$\begin{aligned}\mathcal{H}(\mathbf{q},\mathbf{p}) &= \sum_{i=1}^{N}\frac{\mathbf{p_i}^2}{2m_i} + \frac{1}{2}\sum_{i=1}^{N}\sum_{j\in N_i}^{N}\frac{1}{2}k(\mathbf{q_i}-\mathbf{q_j})^2,\\ \mathcal{H}(\mathbf{q},-\mathbf{p}) &= \sum_{i=1}^{N}\frac{(-\mathbf{p_i})^2}{2m_i} + \frac{1}{2}\sum_{i=1}^{N}\sum_{j\in N_i}^{N}\frac{1}{2}k(\mathbf{q_i}-\mathbf{q_j})^2,\end{aligned} \tag{30}$$

It is obvious $\mathcal{H}(\mathbf{q},\mathbf{p}) = \mathcal{H}(\mathbf{q},-\mathbf{p})$, so it is reversible

### B.2 NON-CONSERVATIVE AND REVERSIBLE SYSTEMS.

A simple example is a n-body spring system with periodical external force, which can be described by:

$$\begin{aligned}\frac{d\mathbf{q_i}}{dt} &= \frac{\partial\mathcal{H}(\mathbf{q},\mathbf{p})}{\partial(\mathbf{p_i})} = \frac{\mathbf{p_i}}{m}\\ \frac{d\mathbf{p_i}}{dt} &= -\frac{\partial\mathcal{H}(\mathbf{q},\mathbf{p})}{\partial(\mathbf{q_i})} = \sum_{j\in N_i}^{N} -k(\mathbf{q_i}-\mathbf{q_j}) - k_1\cos\omega t,\end{aligned} \tag{31}$$

The Hamilton's equations are:

$$\begin{aligned}\frac{\partial\mathcal{H}(\mathbf{q},\mathbf{p})}{\partial\mathbf{p_i}} &= \frac{d\mathbf{q_i}}{dt} = \frac{\mathbf{p_i}}{m}\\ \frac{\partial\mathcal{H}(\mathbf{q},\mathbf{p})}{\partial\mathbf{q_i}} &= -\frac{d\mathbf{p_i}}{dt} = \sum_{j\in N_i} k(\mathbf{q_i}-\mathbf{q_j}) + k_1\cos\omega t,\end{aligned} \tag{32}$$

Hence, we can obtain the Hamiltonian through the integration of the above equation:

$$\mathcal{H}(\mathbf{q},\mathbf{p}) = \sum_{i=1}^{N}\frac{\mathbf{p_i}^2}{2m_i} + \frac{1}{2}\sum_{i=1}^{N}\sum_{j\in N_i}^{N}\frac{1}{2}k(\mathbf{q_i}-\mathbf{q_j})^2 + \sum_{i=1}^{N} q_i * k_1\cos\omega t, \tag{33}$$

**Verify the systems' energy conservation**

$$\begin{aligned}\frac{d\mathcal{H}(\mathbf{q},\mathbf{p})}{dt} =& \frac{1}{dt}(\sum_{i=1}^{N}\frac{\mathbf{p_i}^2}{2m_i}) + \frac{1}{dt}\big(\frac{1}{2}\sum_{i=1}^{N}\sum_{j\in N_i}^{N}\frac{1}{2}k(\mathbf{q_i}-\mathbf{q_j})^2\big) + \frac{1}{dt}\big(\sum_{i=1}^{N} q_i * k_1\cos\omega t\big)\\ =& 0 + \frac{1}{dt}\big(\sum_{i=1}^{N} q_i k_1\cos\omega t\big)\\ =& \big(\sum_{i=1}^{N} -\omega q_i k_1 \sin\omega t\big) \neq 0\end{aligned} \tag{34}$$

So it is non-conservative.

**Verify the systems' time-reversal symmetry** We do the transformation $R : (\mathbf{q},\mathbf{p},t) \mapsto (\mathbf{q},-\mathbf{p},-t)$.

$$\begin{aligned}\mathcal{H}(\mathbf{q},\mathbf{p}) &= \sum_{i=1}^{N}\frac{\mathbf{p_i}^2}{2m_i} + \frac{1}{2}\sum_{i=1}^{N}\sum_{j\in N_i}^{N}\frac{1}{2}k(\mathbf{q_i}-\mathbf{q_j})^2 + \sum_{i=1}^{N} q_i * k_1\cos\omega t,\\ \mathcal{H}(\mathbf{q},-\mathbf{p}) &= \sum_{i=1}^{N}\frac{(-\mathbf{p_i})^2}{2m_i} + \frac{1}{2}\sum_{i=1}^{N}\sum_{j\in N_i}^{N}\frac{1}{2}k(\mathbf{q_i}-\mathbf{q_j})^2 + \sum_{i=1}^{N} q_i * k_1\cos\omega(-t),\end{aligned} \tag{35}$$

It is obvious $\mathcal{H}(\mathbf{q},\mathbf{p},t) = \mathcal{H}(\mathbf{q},-\mathbf{p},t)$, so it is reversible

### B.3 NON-CONSERVATIVE AND IRREVERSIBLE SYSTEMS.

A simple example is an n-body spring system with frictions proportional to its velocity,$\gamma$ is the coefficient of friction, which can be described by:

$$\begin{aligned} \frac{d\mathbf{q}_i}{dt} &= \frac{\partial \mathcal{H}(\mathbf{q},\mathbf{p})}{\partial \mathbf{p}_i} = \frac{\mathbf{p}_i}{m} \\ \frac{d\mathbf{p}_i}{dt} &= -\frac{\partial \mathcal{H}(\mathbf{q},\mathbf{p})}{\partial \mathbf{q}_i} = -k_0\mathbf{q}_i - \gamma\frac{\mathbf{p}_i}{m} \end{aligned} \tag{36}$$

The Hamilton's equations are:

$$\begin{aligned} \frac{\partial \mathcal{H}(\mathbf{q},\mathbf{p})}{\partial \mathbf{p_i}} &= \frac{d\mathbf{q_i}}{dt} = \frac{\mathbf{p_i}}{m} \\ \frac{\partial \mathcal{H}(\mathbf{q},\mathbf{p})}{\partial \mathbf{q_i}} &= -\frac{d\mathbf{p_i}}{dt} = \sum_{j\in N_i} k(\mathbf{q_i}-\mathbf{q_j}) + \gamma\frac{\mathbf{p}_i}{m} \end{aligned} \tag{37}$$

Hence, we can obtain the Hamiltonian through the integration of the above equation:

$$\mathcal{H}(\mathbf{q},\mathbf{p}) = \sum_{i=1}^{N}\frac{\mathbf{p_i}^2}{2m_i} + \frac{1}{2}\sum_{i=1}^{N}\sum_{j\in N_i}^{N}\frac{1}{2}k(\mathbf{q_i}-\mathbf{q_j})^2 + \sum_{i=1}^{N}\frac{\gamma}{m}\int_0^t \frac{\mathbf{p_i}^2}{m}dt, \tag{38}$$

**Verify the systems' energy conservation**

$$\begin{aligned} &\frac{d\mathcal{H}(\mathbf{q},\mathbf{p})}{dt} \\ =&\frac{1}{dt}(\sum_{i=1}^{N}\frac{\mathbf{p_i}^2}{2m_i}) + \frac{1}{dt}(\frac{1}{2}\sum_{i=1}^{N}\sum_{j\in N_i}^{N}\frac{1}{2}k(\mathbf{q_i}-\mathbf{q_j})^2) + \frac{1}{dt}(\sum_{i=1}^{N}\frac{\gamma}{m}\int_0^t \frac{\mathbf{p_i}^2}{m}dt) \\ =&0 + \frac{1}{dt}(\sum_{i=1}^{N}\frac{\gamma}{m}\int_0^t \frac{\mathbf{p_i}^2}{m}dt) \\ =&(\sum_{i=1}^{N}\frac{\gamma}{m}\frac{\mathbf{p_i}^2}{m}) \neq 0 \end{aligned} \tag{39}$$

So it is non-conservative.

**Verify the systems' time-reversal symmetry** We do the transformation $R : (\mathbf{q},\mathbf{p},t) \mapsto (\mathbf{q},-\mathbf{p},-t)$.

$$\begin{aligned} \mathcal{H}(\mathbf{q},\mathbf{p}) &= \sum_{i=1}^{N}\frac{\mathbf{p_i}^2}{2m_i} + \frac{1}{2}\sum_{i=1}^{N}\sum_{j\in N_i}^{N}\frac{1}{2}k(\mathbf{q_i}-\mathbf{q_j})^2 + \sum_{i=1}^{N}\frac{\gamma}{m}\int_0^t \frac{\mathbf{p_i}^2}{m}dt, \\ \mathcal{H}(\mathbf{q},-\mathbf{p}) &= \sum_{i=1}^{N}\frac{(-\mathbf{p_i})^2}{2m_i} + \frac{1}{2}\sum_{i=1}^{N}\sum_{j\in N_i}^{N}\frac{1}{2}k(\mathbf{q_i}-\mathbf{q_j})^2 + \sum_{i=1}^{N}\frac{\gamma}{m}\int_0^{(-t)} \frac{\mathbf{p_i}^2}{m}d(-t), \end{aligned} \tag{40}$$

It is obvious $\mathcal{H}(\mathbf{q},\mathbf{p},t) \neq \mathcal{H}(\mathbf{q},-\mathbf{p},t)$, so it is irreversible

## C DATASET

In our experiments, all datasets are synthesized from ground-truth physical law via sumulation. We generate four simulated datasets: three $n$-body spring systems under damping, periodic, or no external force, and one chaotic tripe pendulum dataset with three sequentially connected stiff sticks that form. We name the first three as *Sipmle Spring*, *Forced Spring*, and *Damped Spring* respectively. All $n$-body spring systems contain 5 interacting balls, with varying connectivities. Each *Pendulum* system contains 3 connected stiff sticks.

For the $n$-body spring system, we randomly sample whether a pair of objects are connected, and model their interaction via forces defined by Hookes law. In the *Damped spring*, the objects have an additional friction force that is opposite to their moving direction and whose magnitude is proportional to their speed. In the *Forced spring*, all objects have the same external force that changes direction periodically. We show in Figure 1(a), the energy variation in both of the *Damped spring* and *Forced spring* is significant.

For the chaotic triple *Pendulum* , the equations governing the motion are inherently nonlinear. Although this system is deterministic, it is also highly sensitive to the initial condition and numerical errors (Shinbrot et al., 1992; Awrejcewicz et al., 2008; Stachowiak & Okada, 2006). This property is often referred to as the "butterfly effect", as depicted in Fig 8. Unlike for $n$-body spring systems, where the forces and equations of motion can be easily articulated, for the *Pendulum*, the explicit forces cannot be directly defined, and the motion of objects can only be described through Lagrangian formulations North (2021), making the modeling highly complex and raising challenges for accurate learning.

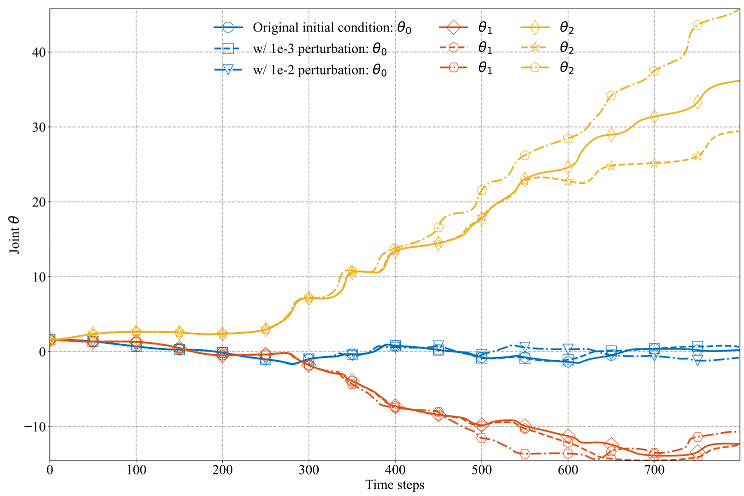


Figure 8: Illustration to show the pendulum is highly-sensitive to initial states

We simulate the trajectories by using Euler's method for $n$-body spring systems and using the 4th order Runge-Kutta (RK4) method for the *Pendulum*. We integrate with a fixed step size and subsample every 100 steps. For training, we use a total of 6000 forward steps. To generate irregularly sampled partial observations, we follow Huang et al. (2020) and sample the number of observations n from a uniform distribution U(40, 52) and draw the n observations uniformly for each object. For testing, we additionally sample 40 observations following the same procedure from PDE steps [6000, 12000], besides generating observations from steps [1, 6000]. The above sampling procedure is conducted independently for each object. We generate 20k training samples and 5k testing samples for each dataset. The features (position/velocity) are normalized to the maximum absolute value of 1 across training and testing datasets.

In the following subsections, we show the dynamical equations of each dataset in detail.

### C.1 SPRING

#### C.1.1 SIMPLE SPRING

The dynamical equations of *simple spring* are as follows:

$$\begin{aligned}\frac{d\mathbf{q_i}}{dt} &= \frac{\mathbf{p_i}}{m} \\ \frac{d\mathbf{p_i}}{dt} &= \sum_{j \in N_i}^{N} -k(\mathbf{q_i} - \mathbf{q_j})\end{aligned} \tag{41}$$

where where $\mathbf{q} = (\mathbf{q_1}, \mathbf{q_2}, \cdots, \mathbf{q_N})$ is a set of positions of each object , $\mathbf{p} = (\mathbf{p_1}, \mathbf{p_2}, \cdots, \mathbf{p_N})$ is a set of momenta of each object. We set the mass of each object $m = 1$, the spring constant$k = 0.1$.

### C.1.2 Damped spring

The dynamical equations of *damped spring* are as follows:

$$\begin{aligned} \frac{d\mathbf{q_i}}{dt} &= \frac{\mathbf{p_i}}{m} \\ \frac{d\mathbf{p_i}}{dt} &= \sum_{j\in N_i} -k(\mathbf{q_i} - \mathbf{q_j}) - \gamma \frac{\mathbf{p}_i}{m} \end{aligned} \tag{42}$$

where where $\mathbf{q} = (\mathbf{q_1}, \mathbf{q_2}, \cdots, \mathbf{q_N})$ is a set of positions of each object, $\mathbf{p} = (\mathbf{p_1}, \mathbf{p_2}, \cdots, \mathbf{p_N})$ is a set of momenta of each object, We set the mass of each object $m = 1$, the spring constant$k = 0.1$, the coefficient of friction $\gamma = 10$.

### C.1.3 Forced spring

The dynamical equations of *forced spring* system are as follows:

$$\begin{aligned} \frac{d\mathbf{q_i}}{dt} &= \frac{\mathbf{p_i}}{m} \\ \frac{d\mathbf{p_i}}{dt} &= \sum_{j\in N_i}^{N} -k(\mathbf{q_i} - \mathbf{q_j}) - k_1 \cos \omega t, \end{aligned} \tag{43}$$

where where $\mathbf{q} = (\mathbf{q_1}, \mathbf{q_2}, \cdots, \mathbf{q_N})$ is a set of positions of each object , $\mathbf{p} = (\mathbf{p_1}, \mathbf{p_2}, \cdots, \mathbf{p_N})$ is a set of momenta of each object. We set the mass of each object $m = 1$ , the spring constant$k = 0.1$, the external strength $k_1 = 10$ and the frequency of variation $\omega = 1$

We simulate the positions and momentums of three spring systems by using Euler methods as follows:

$$\begin{aligned} \mathbf{q_i}(t+1) &= \mathbf{q_i}(t) + \frac{d\mathbf{q_i}}{dt}\Delta t \\ \mathbf{p_i}(t+1) &= \mathbf{p_i}(t) + \frac{d\mathbf{p_i}}{dt}\Delta t \end{aligned} \tag{44}$$

where $\frac{d\mathbf{q_i}}{dt}$ and $\frac{d\mathbf{p_i}}{dt}$ were defined as above for each datasets, and $\Delta t = 0.001$ is the integration steps.

## C.2 CHAOTIC PENDULUM

In this section, we demonstrate how to derive the dynamics equations for a chaotic triple pendulum using the Lagrangian formalism.

The moment of inertia of each stick about the centroid is

$$I = \frac{1}{12} m l^2 \tag{45}$$

The position of the center of gravity of each stick is as follows:

$$\begin{aligned} &x_1 = \frac{l}{2}\sin\theta_1, \quad y_1 = -\frac{l}{2}\cos\theta_1 \\ &x_2 = l(\sin\theta_1 + \frac{1}{2}\sin\theta_2), \quad y_2 = -l(\cos\theta_1 + \frac{1}{2}\cos\theta_2) \\ &x_3 = l(\sin\theta_1 + \sin\theta_2 + \frac{1}{2}\sin\theta_3), \quad y_3 = -l(\cos\theta_1 + \cos\theta_2 + \frac{1}{2}\cos\theta_3) \end{aligned} \tag{46}$$

The change in the center of gravity of each stick is:

$$\begin{aligned} &\dot{x}_1 = \frac{l}{2}\cos\theta_1 \cdot \dot{\theta}_1, \quad \dot{y}_1 = \frac{l}{2}\sin\theta_1 \cdot \dot{\theta}_1 \\ &\dot{x}_2 = l(\cos\theta_1 \cdot \dot{\theta}_1 + \frac{1}{2}\cos\theta_2 \cdot \dot{\theta}_2), \quad \dot{y}_2 = l(\sin\theta_1 \cdot \dot{\theta}_1 + \frac{1}{2}\sin\theta_2 \cdot \dot{\theta}_2) \\ &\dot{x}_3 = l(\cos\theta_1 \cdot \dot{\theta}_1 + \cos\theta_2 \cdot \dot{\theta}_2 + \frac{1}{2}\cos\theta_3 \cdot \dot{\theta}_3), \quad \dot{y}_3 = l(\sin\theta_1 \cdot \dot{\theta}_1 + \sin\theta_2 \cdot \dot{\theta}_2 + \frac{1}{2}\sin\theta_3 \cdot \dot{\theta}_3) \end{aligned} \tag{47}$$

The Lagrangian L of this triple pendulum system is:

$$\begin{aligned}\mathcal{L} =& T - V \\ =& \frac{1}{2}m(\dot{x_1}^2 + \dot{x_2}^2 + \dot{x_3}^2 + \dot{y_1}^2 + \dot{y_2}^2 + \dot{y_3}^2) + \frac{1}{2}I(\dot{\theta_1}^2 + \dot{\theta_2}^2 + \dot{\theta_3}^2) - mg(y_1 + y_2 + y_3) \\ =& \frac{1}{6}ml(9\dot{\theta}_2\dot{\theta}_1 l\cos(\theta_1 - \theta_2) + 3\dot{\theta}_3\dot{\theta}_1 l\cos(\theta_1 - \theta_3) + 3\dot{\theta}_2\dot{\theta}_3 l\cos(\theta_2 - \theta_3) + 7\dot{\theta}_1^2 l + 4\dot{\theta}_2^2 l + \dot{\theta}_3^2 l \\ & + 15g\cos(\theta_1) + 9g\cos(\theta_2) + 3g\cos(\theta_3))\end{aligned} \tag{48}$$

The Lagrangian equation is defined as follows:

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{\theta}} - \frac{\partial \mathcal{L}}{\partial \theta} = \mathbf{0} \tag{49}$$

and we also have:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \dot{\theta}} &= \frac{\partial T}{\partial \dot{\theta}} = p \\ \dot{p} &= \frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{\theta}} = \frac{\partial \mathcal{L}}{\partial \theta}\end{aligned} \tag{50}$$

where p is the Angular Momentum.
We can list the equations for each of the three sticks separately:

$$\begin{aligned}p_1 = \frac{\partial \mathcal{L}}{\partial \dot{\theta_\mathbf{1}}} \quad \dot{p_1} = \frac{\partial \mathcal{L}}{\partial \theta_\mathbf{1}} \\ p_2 = \frac{\partial \mathcal{L}}{\partial \dot{\theta_\mathbf{2}}} \quad \dot{p_2} = \frac{\partial \mathcal{L}}{\partial \theta_\mathbf{2}} \\ p_3 = \frac{\partial \mathcal{L}}{\partial \dot{\theta_\mathbf{3}}} \quad \dot{p_3} = \frac{\partial \mathcal{L}}{\partial \theta_\mathbf{3}}\end{aligned} \tag{51}$$

Finally, we have :

$$\begin{cases} \dot{\theta_1} = \frac{6(9p_1\cos(2(\theta_2-\theta_3))+27p_2\cos(\theta_1-\theta_2)-9p_2\cos(\theta_1+\theta_2-2\theta_3)+21p_3\cos(\theta_1-\theta_3)-27p_3\cos(\theta_1-2\theta_2+\theta_3)-23p_1)}{ml^2(81\cos(2(\theta_1-\theta_2))-9\cos(2(\theta_1-\theta_3))+45\cos(2(\theta_2-\theta_3))-169)} \\ \dot{\theta_2} = \frac{6(27p_1\cos(\theta_1-\theta_2)-9p_1\cos(\theta_1+\theta_2-2\theta_3)+9p_2\cos(2(\theta_1-\theta_3))-27p_3\cos(2\theta_1-\theta_2-\theta_3)+57p_3\cos(\theta_2-\theta_3)-47p_2)}{ml^2(81\cos(2(\theta_1-\theta_2))-9\cos(2(\theta_1-\theta_3))+45\cos(2(\theta_2-\theta_3))-169)} \\ \dot{\theta_3} = \frac{6(21p_1\cos(\theta_1-\theta_3)-27p_1\cos(\theta_1-2\theta_2+\theta_3)-27p_2\cos(2\theta_1-\theta_2-\theta_3)+57p_2\cos(\theta_2-\theta_3)+81p_3\cos(2(\theta_1-\theta_2))-143p_3)}{ml^2(81\cos(2(\theta_1-\theta_2))-9\cos(2(\theta_1-\theta_3))+45\cos(2(\theta_2-\theta_3))-169)} \\ \dot{p_1} = -\frac{1}{2}ml\left(3\dot{\theta}_2\dot{\theta}_1 l\sin(\theta_1-\theta_2) + \dot{\theta}_1\dot{\theta}_3 l\sin(\theta_1-\theta_3) + 5g\sin(\theta_1)\right) \\ \dot{p_1} = -\frac{1}{2}ml\left(-3\dot{\theta}_1\dot{\theta}_2 l\sin(\theta_1-\theta_2) + \dot{\theta}_2\dot{\theta}_3 l\sin(\theta_2-\theta_3) + 3g\sin(\theta_2)\right) \\ \dot{p_1} = -\frac{1}{2}ml\left(\dot{\theta}_1\dot{\theta}_3 l\sin(\theta_1-\theta_3) + \dot{\theta}_2\dot{\theta}_3 l\sin(\theta_2-\theta_3) - g\sin(\theta_3)\right) \end{cases} \tag{52}$$

We simulate the angular of the three sticks by using the Runge-Kutta 4th Order Method as follows:

$$\begin{aligned}\Delta\boldsymbol{\theta}^1(t) &= \dot{\boldsymbol{\theta}}(t, \boldsymbol{\theta}(t)) \cdot \Delta t \\ \Delta\boldsymbol{\theta}^2(t) &= \dot{\boldsymbol{\theta}}(t + \frac{\Delta t}{2}, \boldsymbol{\theta}(t) + \frac{\Delta\boldsymbol{\theta}^1(t)}{2}) \cdot \Delta t \\ \Delta\boldsymbol{\theta}^3(t) &= \dot{\boldsymbol{\theta}}(t + \frac{\Delta t}{2}, \boldsymbol{\theta}(t) + \frac{\Delta\boldsymbol{\theta}^2(t)}{2}) \cdot \Delta t \\ \Delta\boldsymbol{\theta}^4(t) &= \dot{\boldsymbol{\theta}}(t + \Delta t, \boldsymbol{\theta}(t) + \Delta\boldsymbol{\theta}^3(t)) \cdot \Delta t \\ \Delta\boldsymbol{\theta}(t) &= \frac{1}{6}(\Delta\boldsymbol{\theta}^1(t) + \Delta\boldsymbol{\theta}^2(t) + \Delta\boldsymbol{\theta}^3(t) + \Delta\boldsymbol{\theta}^4(t)) \\ \boldsymbol{\theta}(t+1) &= \boldsymbol{\theta}(t) + \Delta\boldsymbol{\theta}(t)\end{aligned} \tag{53}$$

where $\dot{\boldsymbol{\theta}}$ was defined as above , and $\Delta t = 0.0001$ is the integration steps.

# D Model Details

In the following we introduce in details how we implement our model and each baseline.

### D.1 Initial State Encoder

The initial state encoder computes the latent node initial states $\boldsymbol{z}_i(t)$ for all agents simultaneously considering their mutual interaction. Specifically, it first fuses all observations into a temporal graph and conducts dynamic node representation through a spatial-temporal GNN as in Huang et al. (2020):

$$\boldsymbol{h}_{j(t)}^{l+1} = \boldsymbol{h}_{j(t)}^{l} + \sigma\left(\sum_{i(t')\in\mathcal{N}_{j(t)}} \alpha_{i(t')\to j(t)}^{l} \times \boldsymbol{W}_v \hat{\boldsymbol{h}}_{i(t')}^{l-1}\right)$$
$$\alpha_{i(t')\to j(t)}^{l} = \left(\boldsymbol{W}_k \hat{\boldsymbol{h}}_{i(t')}^{l-1}\right)^T \left(\boldsymbol{W}_q \boldsymbol{h}_{j(t)}^{l-1}\right) \cdot \frac{1}{\sqrt{d}}, \quad \hat{\boldsymbol{h}}_{i(t')}^{l-1} = \boldsymbol{h}_{i(t')}^{l-1} + \mathrm{TE}(t'-t) \tag{54}$$
$$\mathrm{TE}(\Delta t)_{2i} = \sin\left(\frac{\Delta t}{10000^{2i/d}}\right), \ \mathrm{TE}(\Delta t)_{2i+1} = \cos\left(\frac{\Delta t}{10000^{2i/d}}\right),$$

where $||$ denotes concatenation; $\sigma(\cdot)$ is a non-linear activation function; $d$ is the dimension of node embeddings. The node representation is computed as a weighted summation over its neighbors plus residual connection where the attention score is a transformer-based Vaswani et al. (2017) dot-product of node representations by the use of value, key, query projection matrices $\boldsymbol{W}_v, \boldsymbol{W}_k, \boldsymbol{W}_q$. Here $\boldsymbol{h}_{j(t)}^{l}$ is the representation of agent $j$ at time $t$ in the $l$-th layer. $i(t')$ is the general index for neighbors connected by temporal edges (where $t' \neq t$) and spatial edges (where $t = t'$ and $i \neq j$). The temporal encoding Hu et al. (2020) is added to a neighborhood node representation in order to distinguish its message delivered via spatial and temporal edges. Then, we stack $L$ layers to get the final representation for each observation node: $\boldsymbol{h}_i^t = \boldsymbol{h}_{i(t)}^{L}$. Finally, we employ a self-attention mechanism to generate the sequence representation $\boldsymbol{u}_i$ for each agent as their latent initial states:

$$\boldsymbol{u}_i = \frac{1}{K}\sum_t \sigma\left(\boldsymbol{a}_i^T \hat{\boldsymbol{h}}_i^t \hat{\boldsymbol{h}}_i^t\right), \ \boldsymbol{a}_i = \tanh\left(\left(\frac{1}{K}\sum_t \hat{\boldsymbol{h}}_i^t\right)\boldsymbol{W}_a\right), \tag{55}$$

where $\boldsymbol{a}_i$ is the average of observation representations with a nonlinear transformation $\boldsymbol{W}_a$ and $\hat{\boldsymbol{h}}_i^t = \boldsymbol{h}_i^t + \mathrm{TE}(t)$. $K$ is the number of observations for each trajectory. Compared with recurrent models such as RNN, LSTM Sepp & Jürgen (1997), it offers better parallelization for accelerating training speed and in the meanwhile alleviates the vanishing/exploding gradient problem brought by long sequences.

Given the latent initial states, the dynamics of the whole system are determined by the ODE function $g$ which we parametrize as a GNN as in Huang et al. (2020) to capture the continuous interaction among agents. We then employ Multilayer Perceptron (MLP) as a decoder to predict the trajectories $\hat{\boldsymbol{y}}_i(t)$ from the latent states $\boldsymbol{z}_i(t)$.

$$\begin{aligned}\boldsymbol{z}_i(t), \boldsymbol{z}_i(1), \boldsymbol{z}_i(2)\cdots\boldsymbol{z}_i(T) &= \mathrm{ODEsolver}(g, [\boldsymbol{z}_1(0), \boldsymbol{z}_2(0)\cdots\boldsymbol{z}_N(0)], (t_0, t_{1T}))\\ \hat{\boldsymbol{y}}_i(t) &= f_{dec}(\boldsymbol{z}_i(t))\end{aligned} \tag{56}$$

### D.2 Implementation Details

**TANGO**

Our implementation of TANGO follows GraphODE pipeline. We implement the initial state encoder using a 2-layer GNN with a hidden dimension of 64 across all datasets. We use ReLU for nonlinear activation. For the sequence self-attention module, we set the output dimension to 128. The encoder's output dimension is set to 16, and we add 64 additional dimensions initialized with all zeros to the latent states $z_i(t)$ to stabilize the training processes as in Huang et al. (2021). The GNN ODE function is implemented with a single-layer GNN from Kipf et al. (2018) with hidden dimension 128. To compute trajectories, we use the Runge-Kutta method from torchdiffeq python package s(Chen et al., 2021) as the ODE solver and a one-layer MLP as the decoder.

We implement our model in pytorch. Encoder, generative model, and the decoder parameters are jointly optimized with AdamW optimizer (Loshchilov & Hutter, 2019) using a learning rate of 0.0001 for spring datasets and 0.00001 for *Pendulum*. The batch size for all datasets is set to 512.

$\text{TANGO}_{\text{gt-rev}}$ and $\text{TANGO}_{\text{rev2}}$ share the same architecture and hyparameters as TANGO, with different implementations of the loss function. In $\text{TANGO}_{\text{gt-rev}}$, instead of comparing forward and reverse trajectories, we look at the L2 distance between the ground truth and reverse trajectories when computing the reversal loss.

For $\text{TANGO}_{\text{rev2}}$, we implement the reversal loss following Huh et al. (2020) with one difference: we do not apply the reverse operation to the momentum portion of the initial state to the ODE function. This is because the initial hidden state is an output of the encoder that mixes position and momentum information. Note that we also remove the additional dimensions to the latent state that TANGO has.

**LatentODE**

We implement the Latent ODE sequence to sequence model as specified in Rubanova et al. (2019). We use a 4-layer ODE function in the recognition ODE, and a 2-layer ODE function in the generative ODE. The recognition and generative ODEs use Euler and Dopri5 as solvers (Chen et al., 2021), respectively. The number of units per layer is 1000 in the ODE functions and 50 in GRU update networks. The dimension of the recognition model is set to 100. The model is trained with a learning rate of 0.001 with an exponential decay rate of 0.999 across different experiments. Note that since latentODE is a single-agent model, we compute the trajectory of each object independently when applying it to multi-agent systems.

**HODEN**

To adapt HODEN, which requires full initial states of all objects, to systems with partial observations, we compute each objects initial state via linear spline interpolation if it is missing. Following the setup in Huh et al. (2020), we have two 2-layer linear networks with Tanh activation in between as ODE functions, in order to model both positions and momenta. Each network has a 1000-unit layer followed by a single-unit layer. The model is trained with a learning rate of 0.00001 using a cosine scheduler.

**TRS-ODEN**

Similar to HODEN, we compute each objects initial state via linear spline interpolation if it is missing. As in Huh et al. (2020), we use a 2-layer linear network with Tanh activation in between as the ODE functions, and the Leapfrog method for solving ODEs. The network has 1000 hidden units and is trained with a learning rate of 0.00001 using a cosine scheduler.

**$\text{TRS-ODEN}_{\text{GNN}}$**

For $\text{TRSODEN}_{\text{GNN}}$, we substitute the ODE function in TRS-ODEN with a GraphODE network. The GraphODE generative model is implemented with a single-layer GNN with hidden dimension 128. As in HODEN and TRS-ODEN, we compute each objects missing initial state via linear spline interpolation and the Leapfrog method for solving ODE. For all datasets, we use 0.5 as the coefficient for the reversal loss in Huh et al. (2020), and 0.0002 as the learning rate under cosine scheduling.

**LGODE**

Our implementation follows Huang et al. (2020) except we remove the Variational Autoencoder (VAE) from the initial state encoder. Instead of using the output from the encoder GNN as the mean and std of the VAE, we directly use it as the latent initial state. We use the same architecture as in TANGO and train the model using an AdamW optimizer with a learning rate of 0.0001 across all datasets.

## E LIMITATIONS

Currently TANGO only incorporates inductive bias from the temporal aspect, while there are still many important properties in the spatial aspect such as translation and rotation equivariance Satorras et al. (2021). Future endeavors that combine biases from both temporal and spatial dimensions could unveil a new frontier in dynamical systems modeling.

## 6 Ethics Statement

TANGO is trained upon physical simulation data (e.g., , spring and pendulum) and implemented by public libraries in PyTorch. During the modeling, we neither introduces any social/ethical bias nor amplify any bias in the data. We do not foresee any direct social consequences or ethical issues.

## 7 Reproducibility

To reproduce our model's results, we provide our code implementation link here. Dataset details can be found in Appendix C and we also provide simulator codes for public use. We also show the implemenmtation details of TANGO and baselines in Apendix D.2.

## References

Jan Awrejcewicz, Grzegorz Kudra, and Grzegorz Wasilewski. Chaotic zones in triple pendulum dynamics observed experimentally and numerically. *Applied Mechanics and Materials*, pp. 1–17, 2008.

Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, and koray kavukcuoglu. Interaction networks for learning about objects, relations and physics. In *Advances in Neural Information Processing Systems*. 2016.

Ricky T. Q. Chen, Brandon Amos, and Maximilian Nickel. Learning neural event functions for ordinary differential equations. *International Conference on Learning Representations*, 2021.

Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, 2018.

Miles Cranmer, Sam Greydanus, Stephan Hoyer, Peter Battaglia, David Spergel, and Shirley Ho. Lagrangian neural networks. *arXiv preprint arXiv:2003.04630*, 2020.

David F. Mayers Endre Süli. *An Introduction to Numerical Analysis*. Cambridge University Press, 2003.

Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. *Advances in Neural Information Processing Systems*, 2019.

Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *IEEE international conference on robotics and automation (ICRA)*, 2017.

Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In *Proceedings of the 2020 World Wide Web Conference*, 2020.

Zijie Huang, Yizhou Sun, and Wei Wang. Learning continuous system dynamics from irregularly-sampled partial observations. In *Advances in Neural Information Processing Systems*, 2020.

Zijie Huang, Yizhou Sun, and Wei Wang. Coupled graph ode for learning interacting system dynamics. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery  Data Mining*, 2021.

In Huh, Eunho Yang, Sung Ju Hwang, and Jinwoo Shin. Time-reversal symmetric ode network. In *Advances in Neural Information Processing Systems*, 2020.

Song Jiang, Zijie Huang, Xiao Luo, and Yizhou Sun. Cf-gode: Continuous-time causal inference for multi-agent dynamical systems. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2023.

Sangtae Kim and Seppo J Karrila. *Microhydrodynamics: principles and selected applications*. Courier Corporation, 2013.

Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. *arXiv preprint arXiv:1802.04687*, 2018.

Jeroen SW Lamb and John AG Roberts. Time-reversal symmetry in dynamical systems: a survey. *Physica D: Nonlinear Phenomena*, pp. 1–39, 1998.

Chengshu Li, Fei Xia, Roberto Martín-Martín, Michael Lingelbach, Sanjana Srivastava, Bokui Shen, Kent Elliott Vainio, Cem Gokmen, Gokul Dharan, Tanish Jain, Andrey Kurenkov, Karen Liu, Hyowon Gweon, Jiajun Wu, Li Fei-Fei, and Silvio Savarese. igibson 2.0: Object-centric simulation for robot learning of everyday household tasks. In *Proceedings of the 5th Conference on Robot Learning*, 2022.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *The International Conference on Learning Representations*, 2019.

Xiao Luo, Jingyang Yuan, Zijie Huang, Huiyu Jiang, Yifang Qin, Wei Ju, Ming Zhang, and Yizhou Sun. HOPE: High-order graph ODE for modeling interacting dynamics. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

Emmy Noether. Invariant variation problems. *Transport theory and statistical physics*, 1(3):186–207, 1971.

Jill North. Formulations of classical mechanics. *Forthcoming in A companion to the philosophy of physics. Routledge*, 2021.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. Learning mesh-based simulation with graph networks. In *International Conference on Learning Representations*, 2021.

Michael Poli, Stefano Massaroli, Junyoung Park, Atsushi Yamashita, Hajime Asama, and Jinkyoo Park. Graph neural ordinary differential equations. *arXiv preprint arXiv:1911.07532*, 2019.

Costas Pozrikidis. Interfacial dynamics for stokes flow. *Journal of Computational Physics*, 169(2): 250–301, 2001.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

John AG Roberts and GRW1173588 Quispel. Chaos and time-reversal symmetry. order and chaos in reversible dynamical systems. *Physics Reports*, 216(2-3):63–177, 1992.

Yulia Rubanova, Ricky TQ Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. In *Advances in Neural Information Processing Systems*, 2019.

Alvaro Sanchez-Gonzalez, Victor Bapst, Kyle Cranmer, and Peter Battaglia. Hamiltonian graph networks with ode integrators. In *Advances in Neural Information Processing Systems*, 2019.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to simulate complex physics with graph networks. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Vıctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E (n) equivariant graph neural networks. In *International conference on machine learning*, pp. 9323–9332. PMLR, 2021.

Michael Schober, Simo Särkkä, and Philipp Hennig. A probabilistic model for the numerical solution of initial value problems. In *Statistics and Computing*, pp. 99–122. 2019.

Hochreiter Sepp and Schmidhuber Jürgen. Long short-term memory. *Neural computation*, 1997.

Troy Shinbrot, Celso Grebogi, Jack Wisdom, and James A Yorke. Chaos in a double pendulum. *American Journal of Physics*, (6):491–499, 1992.

Tomasz Stachowiak and Toshio Okada. A numerical analysis of chaos in the double pendulum. *Chaos, Solitons & Fractals*, (2):417–422, 2006.

E. C. Tolman. The determiners of behavior at a choice point. *Psychological Review*, 45(1):1–41, 1938.

Riccardo Valperga, Kevin Webster, Dmitry Turaev, Victoria Klein, and Jeroen Lamb. Learning reversible symplectic dynamics. In *Proceedings of The 4th Annual Learning for Dynamics and Control Conference*, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*. 2017.

Rui Wang, Karthik Kashinath, Mustafa Mustafa, Adrian Albert, and Rose Yu. Towards physics-informed deep learning for turbulent flow prediction. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020.

Song Wen, Hao Wang, and Dimitris Metaxas. Social ode: Multi-agent trajectory forecasting with neural ordinary differential equations. In *European Conference on Computer Vision*, 2022.

Chengxi Zang and Fei Wang. Neural dynamics on complex networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020.