A Dataset specification

A.1 Types of systems used

Below are the descriptions of the systems that are used for the dataset.

Mass-Spring One of the most simplest physical system is one particle, frictionless mass-spring system, where m, k are the mass of the particle, and spring constant respectively.

$$\mathcal{H} = \frac{p_x^2}{2m} + \frac{kx^2}{2}$$

Pendulum The Hamiltonian of a pendulum system is slightly more complex than the mass-spring case. Here, m, g, l denotes the mass of the particle, gravitational acceleration, and length of the pendulum respectively.

$$\mathcal{H} = \frac{p_\theta^2}{2ml^2} + mgl(1 - \cos\theta)$$

Hénon-Heiles Up to here, the given dynamical system was rather simple. Hénon-Heiles gives the chaotic dynamics of a star around a galactic center with its motion constrained on a 2D plane (Hénon & Heiles, 1964; Hénon, 1983). In the below Hamiltonian of the Hénon-Heiles system, λ is conventionally taken as unity.

$$\mathcal{H} = \frac{1}{2}(p_x^2 + p_y^2) + \frac{1}{2}(x^2 + y^2) + \lambda\left(x^2 y - \frac{y^3}{3}\right)$$

Magnetic-Mirror Here we introduce a system that has the most complicated form of Hamiltonian in our experiments. From the works of (Efthymiopoulos et al., 2015), the Hamiltonian of the magnetic bottle type system is given as follows.

$$\mathcal{H} = \frac{1}{2}(\dot{\rho}^2 + \dot{z}^2) + \frac{1}{2}\rho^2 + \frac{1}{2}\rho^2 z^2 - \frac{1}{8}\rho^4 + \frac{1}{8}\rho^2 z^4 - \frac{1}{16}\rho^4 z^2 + \frac{1}{128}\rho^6$$

Two-Body From now on, we expand our system with more than two particles. In the two-body system case, we consider the gravitational interaction between two particles. Then the Hamiltonian of the two-body system can be written as follows. Note that G is the gravitational constant, and m_1, m_2 are the masses for each of the two bodies.

$$\mathcal{H} = \frac{\boldsymbol{p}_1^2}{2m_1} + \frac{\boldsymbol{p}_2^2}{2m_2} - \frac{Gm_1m_2}{|\boldsymbol{r}_1 - \boldsymbol{r}_2|}$$

Three-Body Adding a particle to the two-body system gives the three-body system. Although the Hamiltonian of the three-body system is an incidental extension of the two-body case, its dynamics cannot be described by a closed-form expression, thus exhibits chaotic behaviour.

$$\mathcal{H} = \frac{\boldsymbol{p}_1^2}{2m_1} + \frac{\boldsymbol{p}_2^2}{2m_2} + \frac{\boldsymbol{p}_3^2}{2m_3} - \frac{Gm_1m_2}{|\boldsymbol{r}_1 - \boldsymbol{r}_2|} - \frac{Gm_2m_3}{|\boldsymbol{r}_2 - \boldsymbol{r}_3|} - \frac{Gm_3m_1}{|\boldsymbol{r}_3 - \boldsymbol{r}_1|}$$

A.2 Dataset generation

Using the Hamiltonian described in Section 3.1, we obtained the state trajectories $(\boldsymbol{q}, \boldsymbol{p})$ by employing the LSODA integrator implemented in SciPy (Virtanen et al., 2020). The trajectories were integrated over the interval $[0, 10]$ with 200 steps. Subsequently, we calculated the corresponding time derivatives $(\dot{\boldsymbol{q}}, \dot{\boldsymbol{p}})$ using JAX (Bradbury et al., 2018) and Equation 1. For simplicity, all the constants from the Hamiltonian (i.e., m_i, k, l_i, g, G) were set to 1. The initial conditions employed for each system are described below.

Mass-Spring The initial state (x_0, p_{x0}) is randomly sampled from a uniform distribution over the interval $[-1, 1]^2$. The redundant coordinate (y_0, p_{y0}) is set to $(0, 0)$ as a fixed value.

Pendulum The initial state $(\theta_0, p_{\theta 0})$ is randomly sampled from a uniform distribution over the interval $\left[-\frac{\pi}{2}, \frac{\pi}{2}\right] \times [-1, 1]$. The redundant coordinate (r_0, p_{r0}) is set to $(1, 0)$ as a fixed value.

Hénon-Heiles The initial state $(x_0, y_0, p_{x0}, p_{y0})$ is randomly sampled from a uniform distribution over the interval $[-1, 1]^4$.

Magnetic-Mirror The initial state $(\rho_0, z_0, p_{\rho 0}, z_0)$ is randomly sampled from a uniform distribution over the interval $[-1, 1]^4$.

Two-Body The initial coordinate for the first body (x_{10}, y_{10}) is randomly sampled from a uniform distribution over the interval $[0.5, 1.5]^2$, and $(p_{x_1 0}, p_{y_1 0})$ is calculated to obtain a nearly-circular orbit. Then, the initial state for the second body $(x_{20}, y_{20}, p_{x_2 0}, p_{y_2 0})$ is set to $(-x_{10}, -y_{10}, -p_{x_1 0}, -p_{y_1 0})$ Then we slightly perturbed p_x, p_y by adding a Gaussian noise (multiplied by a constant of 0.1) to the velocity of both two bodies. Here, the velocities are equivalent to the canonical momentum p_x and p_y, as we assume the masses m_1 and m_2 to be equal to 1.

Three-Body The initial state for the three-body system is obtained in a similar way to that of the two-body, except the initial coordinate for the first body (x_{10}, y_{10}) is randomly sampled from a uniform distribution over the interval $[0.8, 1.2]^2$. Again, $(p_{x_1 0}, p_{y_1 0})$ is set to obtain a nearly-circular orbit. The initial state for the second and third body is obtained by rotating the first and second body each by an angle of $\frac{2\pi}{3}$. Here, the Gaussian noise term that is added to each of the bodies is multiplied by a constant of 0.05.

Following the above description, we generated 10000 trajectories for each system.

A.3 Dataset configuration for meta training

The composition of the system types used in our meta training scenario is listed in Table 1.

Table 1: Dataset configuration for meta learning Hamiltonian systems

system to test	systems for meta training
mass-spring	pendulum, Hénon-Heiles, magnetic-mirror
pendulum	mass-spring, Hénon-Heiles, magnetic-mirror
Hénon-Heiles	mass-spring, pendulum, magnetic-mirror
magnetic-mirror	mass-spring, pendulum, Hénon-Heiles
two-body	mass-spring, pendulum, Hénon-Heiles, magnetic-mirror
three-body	mass-spring, pendulum, Hénon-Heiles, magnetic-mirror

B Neural network implementation

B.1 Network architecture

In our experiments, we constructed our model as follows: GCNConv(4, 200) - Mish - GCNConv(200, 200) - Mish - GCNConv(200, 4) - Mish - GlobalMeanPool - Linear(4, 200) - Mish - Linear(200, 200) - Mish - Linear(200, 1) - Mish, where GCNConv and Linear correspond to the graph convolutional layer and the fully connected layer implemented in PyTorch respectively. Thus, the model outputs a single-dimensional scalar value. After the forward pass, the derivative of the output with respect to the input is computed to obtain the time-derivative of the input state, utilizing Equation 1. Note that we did not heavily tune the hyperparameters regarding the network architecture, because we focus on the difference between the meta-trained, pre-trained, and random-initialized model, not obtaining state-of-the-art results.

B.2 Training process

Mass-Spring For both meta-training and pre-training, we used a learning rate of $\alpha = 0.001$ for the gradient step on the inner loop, and the Adam optimizer on the outer loop with a learning rate of $\beta = 0.0005$. The inner gradient update was 1 step, with a total of 5000 iterations on the outer loop. The number of task batches is set to 10, and the number of phase points used for each task was 50 (i.e. among the 200 points in each trajectory, 50 points were randomly sampled for meta training). For evaluation, we used the Adam optimizer with a learning rate of 0.0001.

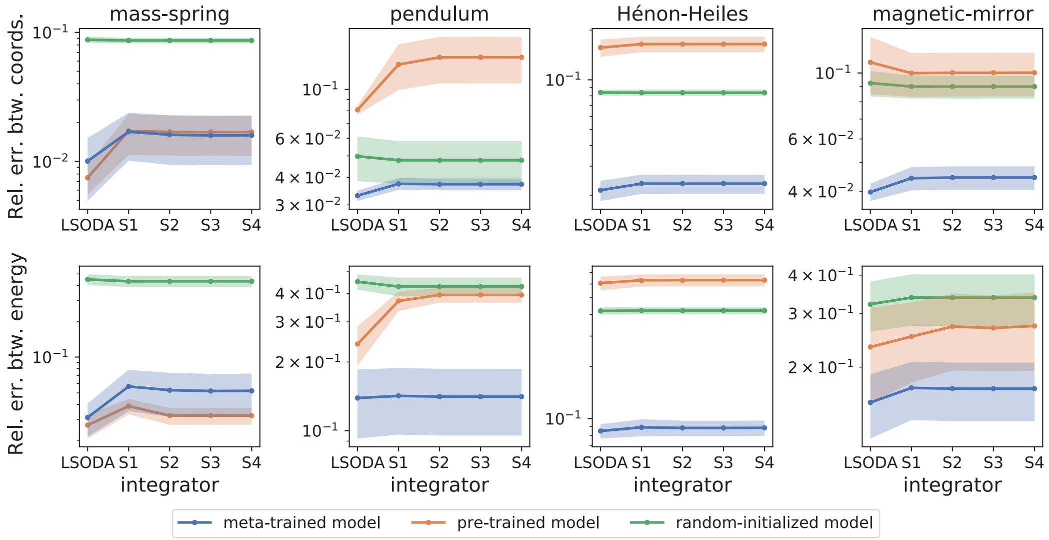

Figure S.1: The relative error for the meta-trained, pre-trained, and random-initialized model with respect to the predicted coordinates (top) and energy (bottom) using different integrators. The solid line and the shaded area each represent the average and the standard error of 10 runs.

Pendulum All conditions were set to be the same as those of the mass-spring system.

Hénon-Heiles All conditions were set to be the same as those of the mass-spring system, except for the iterations on the outer loop which was set to 10000.

Magnetic-Mirror All conditions were set to be the same as those of the mass-spring system, except for the iterations on the outer loop which was set to 30000.

Two-Body For both meta-training and pre-training, we used a learning rate of $\alpha = 0.001$ for the gradient step on the inner loop, and the Adam optimizer on the outer loop with a learning rate of $\beta = 0.0005$. We set 10000 iterations on the outer loop. The other conditions were set to be the same.

Three-Body All conditions were set to be the same as those of the two-body system, except for the β which is set to 0.0005.

C Additional experiments

C.1 Ablation on integrators

Figure S.1 shows the evaluation performance of meta-trained, pre-trained, and randomly-initialized models across scenarios using different integrators; LSODA, and the 1st~4th order symplectic integrator. The LSODA integrator consistently yields the best results in most cases. While symplectic integrators are known for their ability to maintain energy conservation, their advantages become more apparent in long-term dynamics. However, the time range considered in this study is not sufficient to fully capture the benefits of symplectic integrators. Nonetheless, their impact could be significant when extended to longer time scales. So in our experiments, we use LSODA as our primary integrator.

C.2 Ablation on K

Figure S.2 presents the evaluation performance of meta-trained, pre-trained, and randomly-initialized models under varying K in the adaptation task. The ablation results indicate that the meta-trained model consistently performs well at low K values across adaptation tasks. Furthermore, its performance markedly improves as the data size K increases, which reflects the efficient adaptability of the meta-trained model.

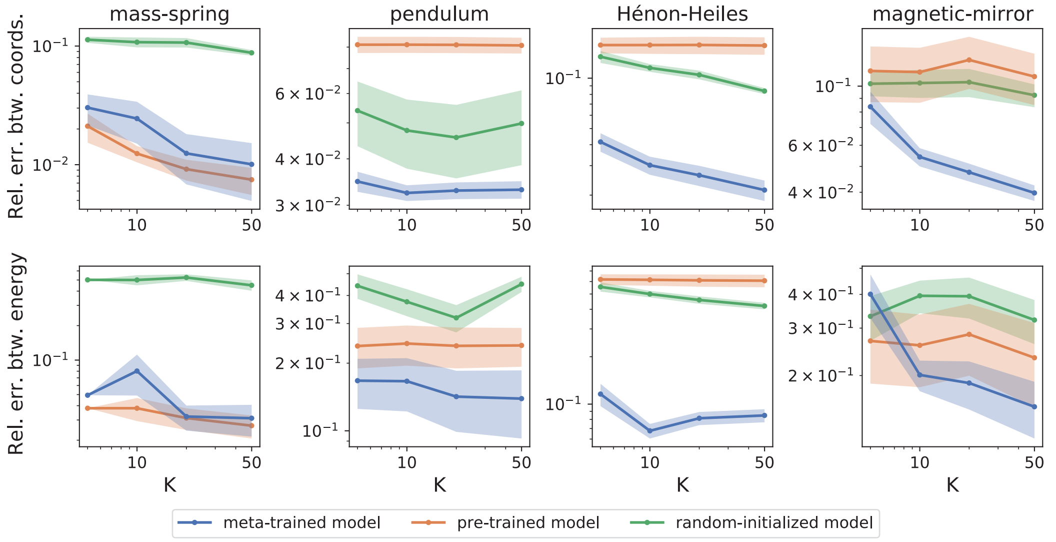

Figure S.2: The relative error for the meta-trained, pre-trained, and random-initialized model with respect to the predicted coordinates (top) and energy (bottom) with various number of data points K used. The solid line and the shaded area each represent the average and the standard error of 10 runs.

C.3 Generalization for multi-held-out system

Here, we present the adaptation performance of the meta-trained models in Section 4 to the multi-held-out system scenario for the extent of the single-held-out scenario demonstrated in Section 4. For example, we performed the adaptation task of the meta-trained model trained with a pendulum, Hénon-Heiles, and the magnetic-mirror system not only for the mass-spring system, but also with the newly introduced 2D harmonic oscillator, and the Kepler system as well.

The relative errors in terms of both predicted coordinates and energy for this test are presented in Figure S.3, and S.4. The results demonstrate that the meta-trained models exhibit proficient adaptation both to the newly introduced 2D harmonic soscillator system and the Kepler system.

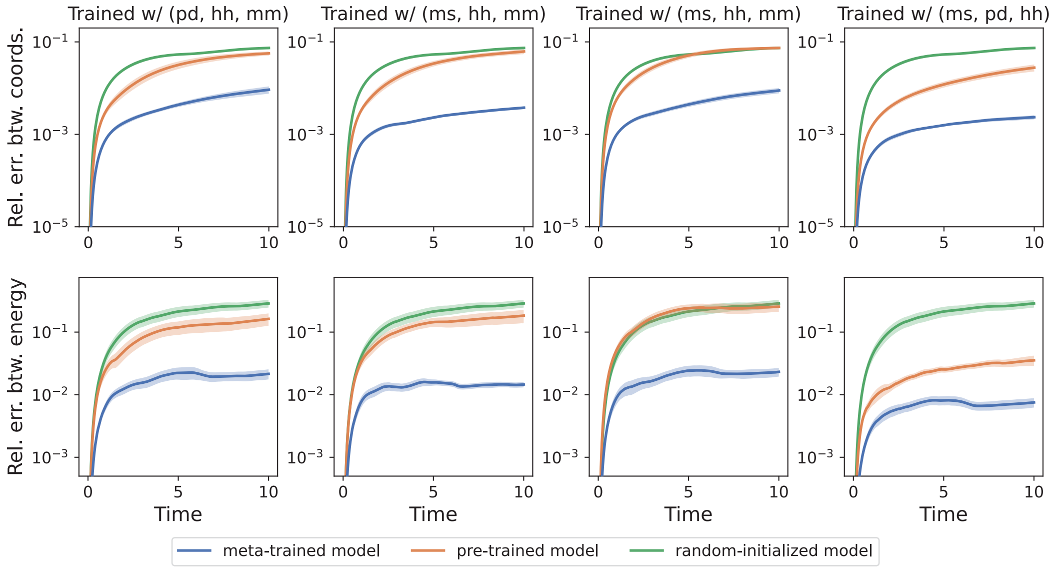

Figure S.3: The relative error throughout the time rollout of the 2D harmonic oscillator system for the meta-trained, pre-trained, and random-initialized model at adaptation step 50 with respect to the predicted coordinates (top) and energy (bottom). The solid line and the shaded area each represent the average and the standard error of 10 runs.

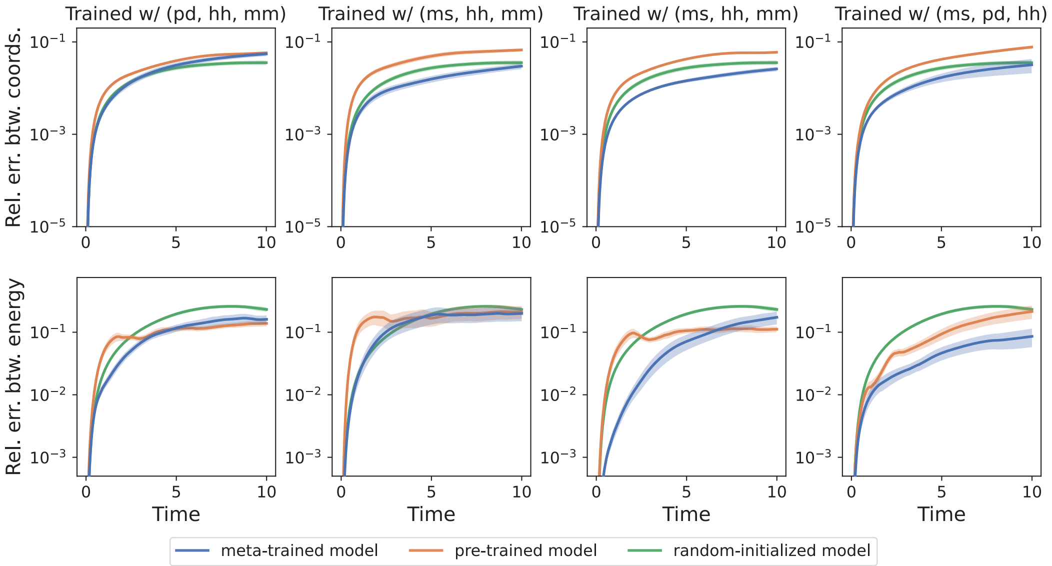

Figure S.4: The relative error throughout the time rollout of the Kepler system for the meta-trained, pre-trained, and random-initialized model at adaptation step 50 with respect to the predicted coordinates (top) and energy (bottom). The solid line and the shaded area each represent the average and the standard error of 10 runs.

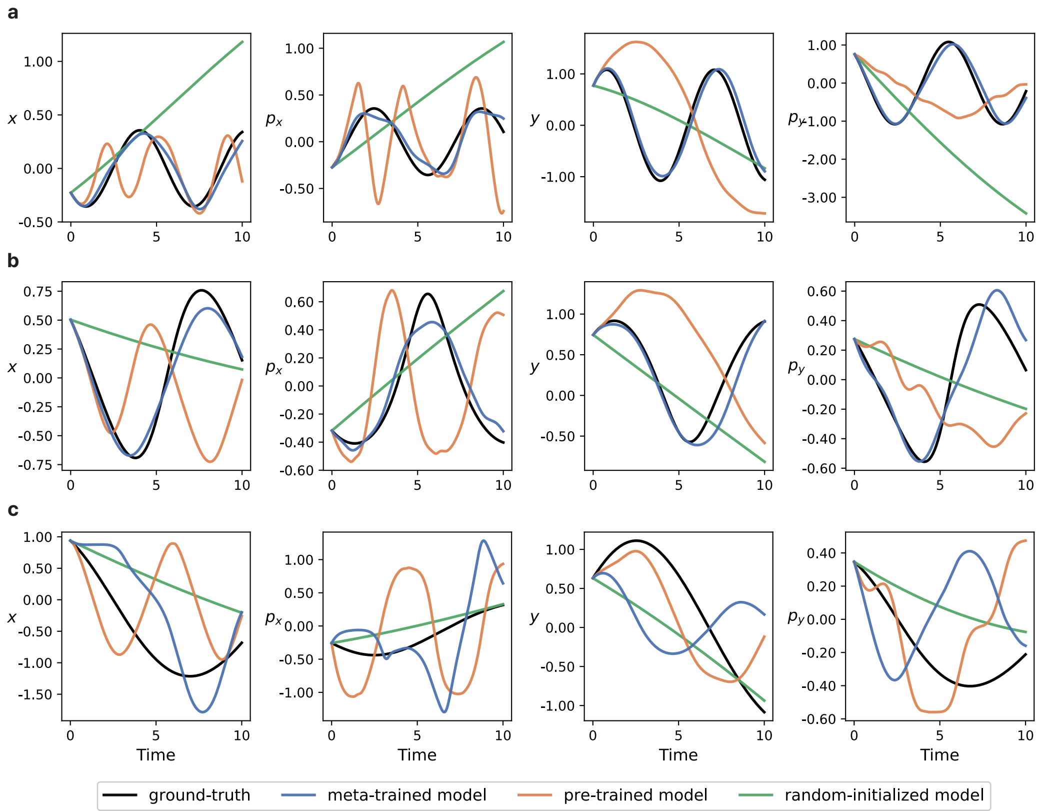

Figure S.5: The example of predicted dynamics for the 2D harmonic oscillator system (a), and each good (b) and bad (c) examples of Kepler system.

We also show the example of the predicted dynamics for both the 2D harmonic oscillator and the Kepler system in Figure S.5. There exists a situation for the Kepler system where the meta-trained model does not perform well, which implies insufficient adaptation steps for this case.

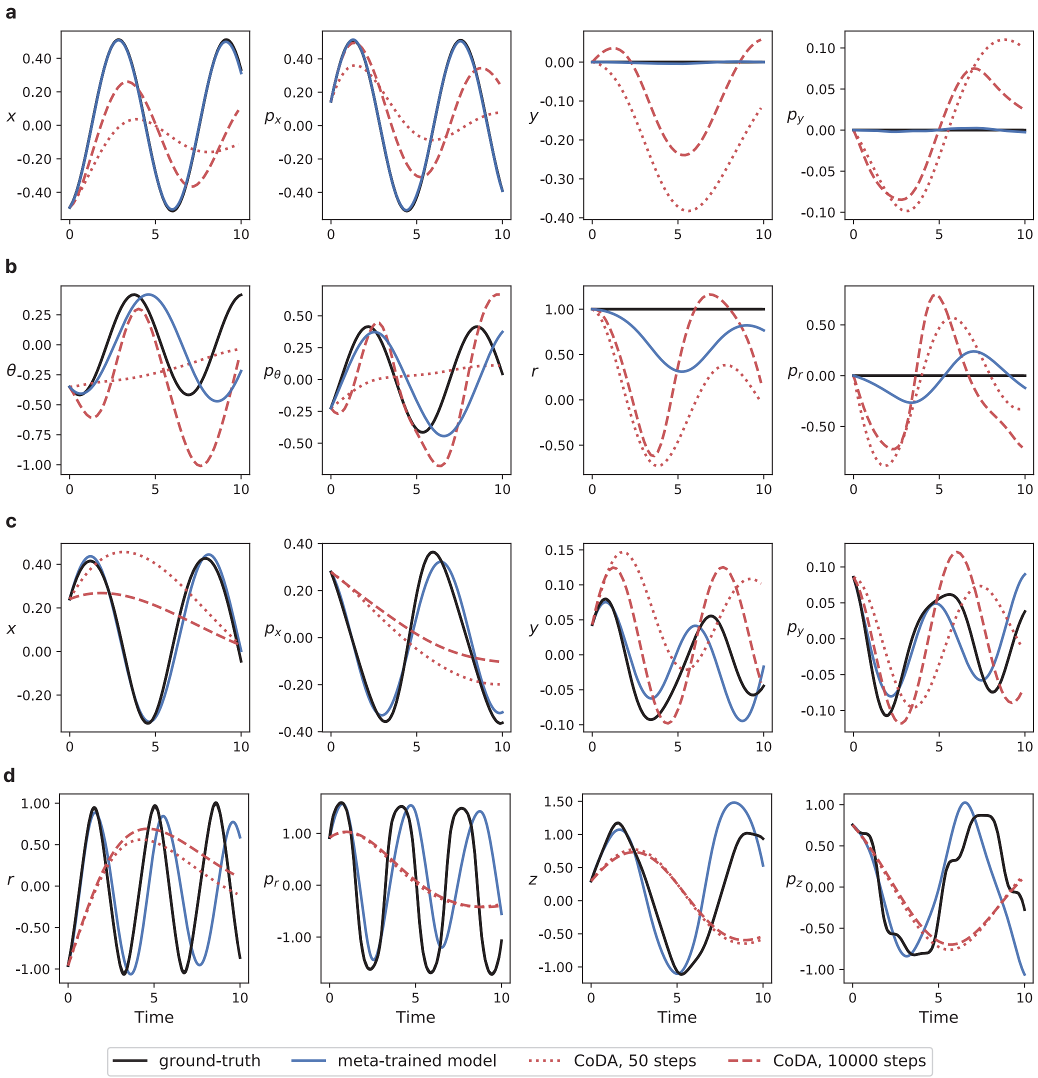

Figure S.6: The prediction of each system dynamics from the meta-trained after 50 adaptation steps, and CoDA after 50 and 10000 steps. Each row (a) to (d) corresponds to the predictions tested with mass-spring, pendulum, Hénon-Heiles, and the magnetic mirror system respectively.

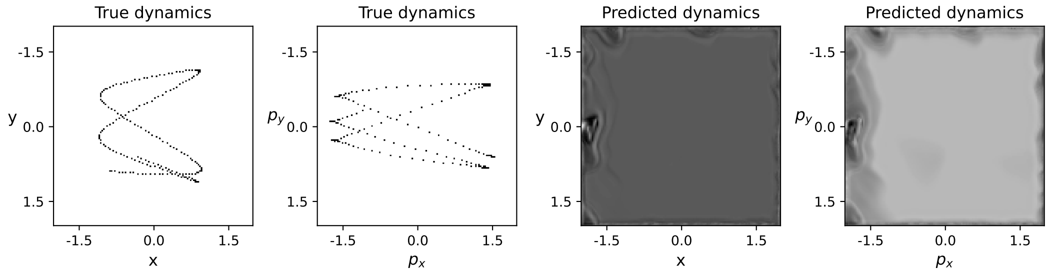

Figure S.7: The predicted dynamics of the magnetic-mirror system using DyAd.

C.4 Comparison with CoDA

The comparison of the meta-trained model between CoDA and DyAd is depicted in Figure S.6. From the results, the dynamics from CoDA after 50 steps (red dotted line) are not sufficient to match the performance of the meta-trained model, so we also made a comparison with the CoDA after 10000 steps (red dashed line), which also failed to meet the expectations.

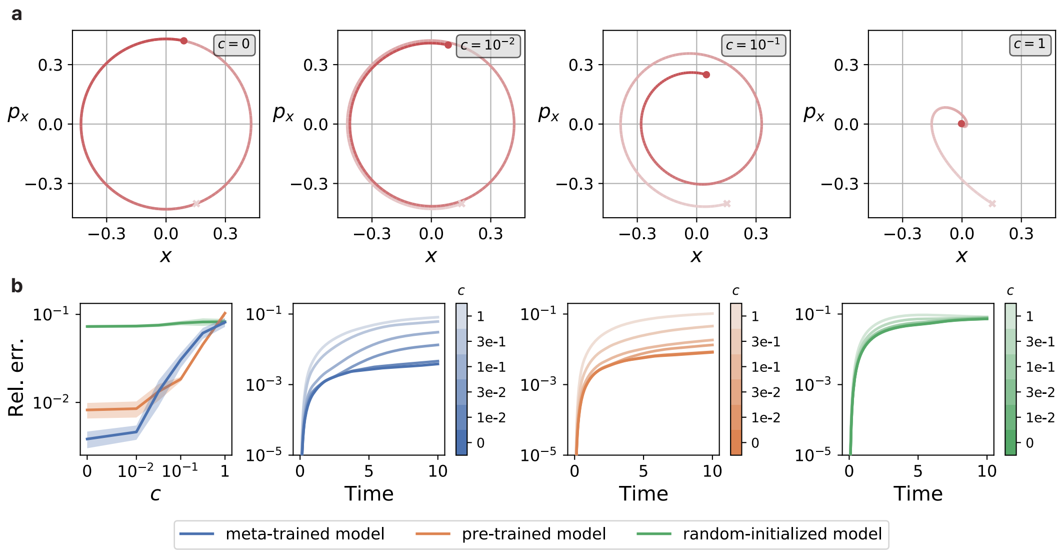

Figure S.8: (a) shows the dissipative dynamics of the mass-spring system as the damping coefficient c changes. The leftmost subfigure in (b) indicates the geometric moving average of the relative error at time 10 for different values of damping coefficient c. The three right subfigures in (b) represent the relative error through the time rollout with different values of c for each three models.

Although CoDA is reported to outperform MAML-based methods in the scope of generalization across environments within a fixed system, our results show that MAML-based method advantages CoDA in our broader scope of definition (i.e. generalization across different types of systems).

C.5 Adaptation with DyAd

Contrary to CoDA, where inputs are represented by the state variables (q, p) of the system, DyAd relies on image sequence inputs. To this end, we generated sequences of images with dimensions $2 \times 128 \times 128$, where each channel corresponds to the xy space and $p_x p_y$ space, respectively.

However, the weak supervision term in DyAd's encoder loss (the first term in Equation 7),

$$\mathcal{L}_{\text{enc}} = \sum_{c \in \mathcal{C}} \|\hat{c} - c\|^2 + \alpha \sum_{i,j,c} \|\hat{z}_c^{(i)} - \hat{z}_c^{(j)}\|^2 + \beta \sum_{i,c} \| \|\hat{z}_c^{(i)}\|^2 - m\|^2 \tag{7}$$

where g is the encoder network, x is the input, $\hat{z}^{(i)} = g(x^{(i)})$, and $\hat{c}^{(i)} = W\hat{z}_c^{(i)} + b$ is an affine transformation of z_c which act as a hidden feature for task c, was not directly applicable to our scenario due to the varying nature of physical parameters across different systems. To adapt this notion to our needs, we have chosen to represent the physical parameter as the energy of the system. While it may not be exact to say that the explicit functional form of energy is common to all systems, the use of energy as a form of weak supervision can effectively utilize the minimal concept of physical parameters across different dynamical systems. We conducted an adaptation task for the magnetic-mirror system using the meta-trained model, which was previously trained on mass-spring, pendulum, and Hénon-Heiles systems. The results show that, as illustrated in Figure S.7, DyAd is not appropriate to our scope of domain generalization.

C.6 Adaptation to dissipative system

Here, we performed an adaptation task to the damped mass-spring system with damping term $-c\dot{x}$ with the same meta-trained model that was used to adapt to the original mass-spring system (i.e. meta-trained with conservative pendulum, Hénon-Heiles, magnetic-mirror system). From Figure S.8, we can see that the meta-trained model with conservative systems cannot easily make an adaptation to a dissipative system, which indicates a definite limitation in our problem setting.

C.7 Demonstration on larger systems

In this section, we present preliminary results for the two-body and three-body tasks. We recognize that the training conditions, such as the learning rate for adaptation, were not sufficiently refined. As a result, while the CKA exhibits behavior similar to that observed in other single-particle tasks as shown in Figure S.10, the prediction accuracy remains suboptimal, as illustrated in Figure S.9. The predicted trajectories are depicted in Figure S.11. It should be noted that while none of the models successfully capture the overall dynamics, the meta-trained model tries to capture the trend than the others. This suggests that with further refinement of the training conditions, the meta-trained model holds promise for achieving more accurate predictions.

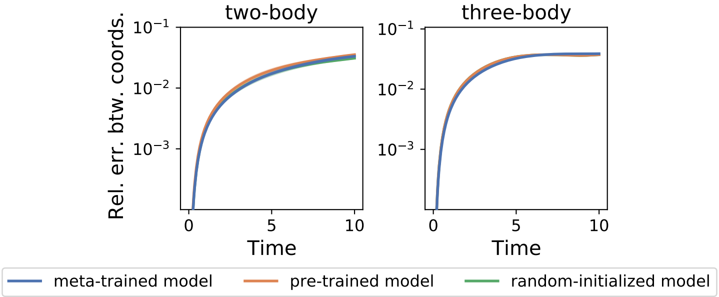

Figure S.9: The relative error throughout time rollout for the meta-trained, pre-trained, and random-initialized model. The solid line and the shaded area each represent the average and the standard error of 10 runs.

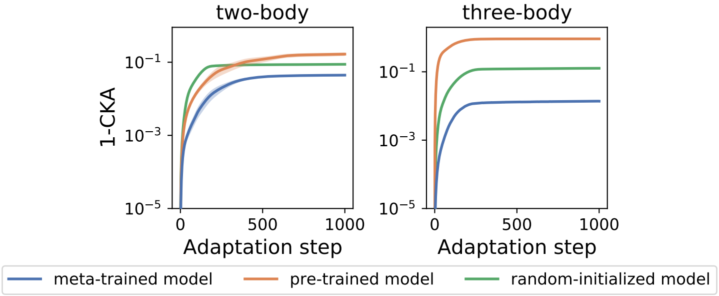

Figure S.10: The measured CKAs in the last layer between the model before making adaptation and during adaptation. The solid line and the shaded area each represent the average and the standard error of 10 runs.

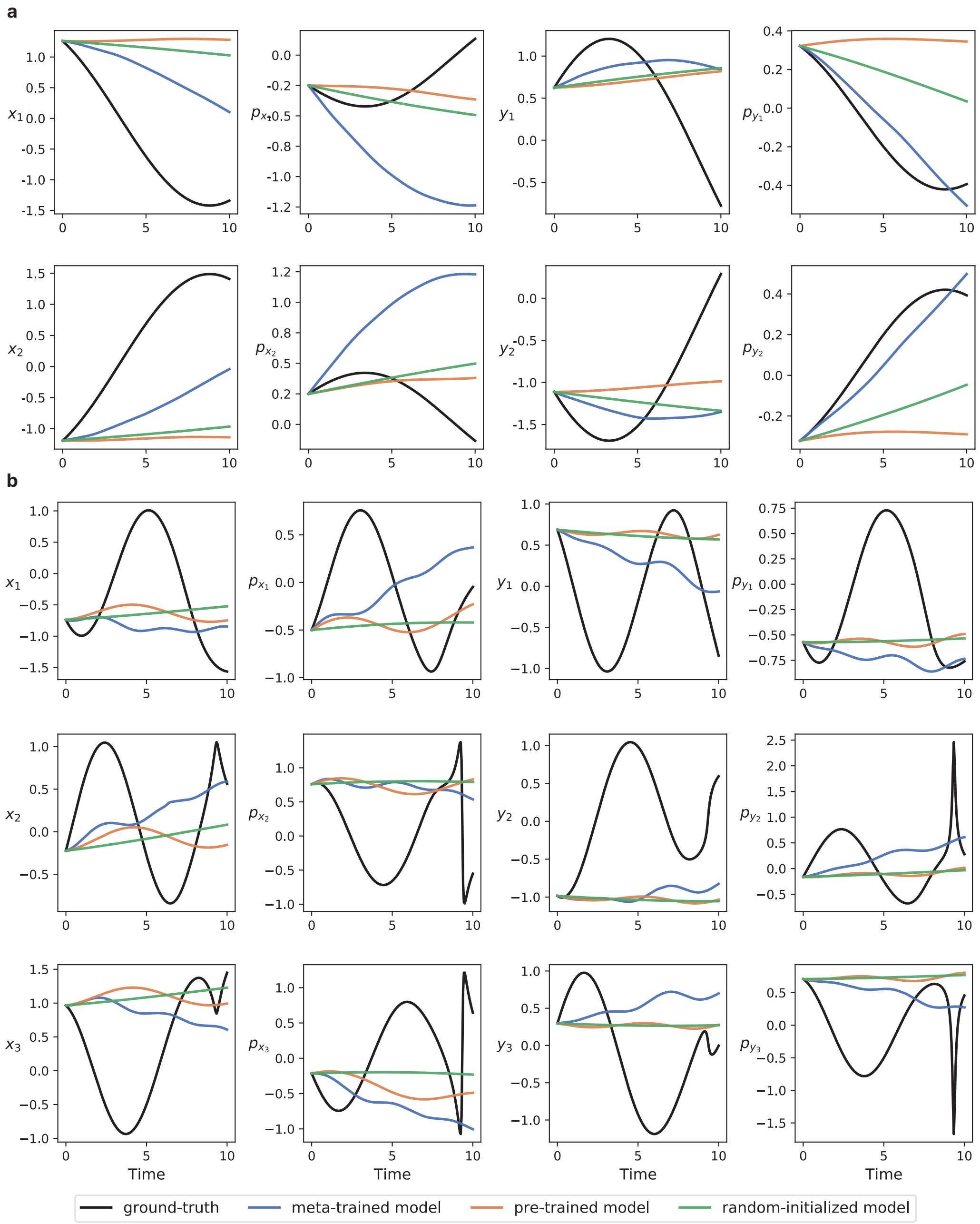

Figure S.11: The example predicted trajectories of (a) two-body, (b) three-body systems.