A Appendix

A.1 Datasets

All of the code used for generating the datasets is included as part of the supplementary material.

A.1.1 Toy Physics

Dataset	Hamiltonian $\mathcal{H}(q,p)$	Hyperparameters
Mass Spring	$k\frac{q^2}{2}+\frac{p^2}{2m}$	k,m
Pendulum	$mlg(1-\cos(q))+\frac{p^2}{2lm}$	m,l,g
Double Pendulum	$\frac{m_2l_2^2p_1^2+(m_1+m_2)l_1^2p_2^2-2m_2l_1l_2p_1p_2cos(q_1-q_2)}{2m_2l_1^2l_2^2(m_1+m_2sin(q_1-q_2)^2}-(m_1+m_2)gl_1cos(q_1)-m_2gl_2cos(q_2)$	m_1,m_2,l_1,l_2,g
N-Body Problem	$-\sum_{i<j}\frac{gm_im_j}{\|\|q_i-q_j\|\|}+\sum_i\frac{\|\|p_i\|\|_2^2}{2m_i}$	g,m_1,...,m_n

Table 2: The Hamiltonians used for simulating all of the classical mechanics systems.

Dataset	Hyperparameters
Mass Spring	$k=2.0$ $m\sim U(0.2,1.0)$
Pendulum	$m\sim U(0.5,1.5)$ $g\sim U(3.0,4.0)$ $l\sim U(0.5,1.0)$
Double Pendulum	$m\sim U(0.4,0.6)$ $g\sim U(2.5,4.0)$ $l\sim U(0.75,1.0)$
Two Body	$m\sim U(0.5,1.5)$ $h\sim U(0.5,1.5)$

Table 3: Sampling protocol for the hyperparameters of the coloured Toy physics datasets.

For simulating the Toy Physics datasets we integrate the Hamiltonian dynamics in a 64-bit precision floating point numbers using the default SciPy integration method `scipy.integrate.solve_ivp`. For systems where there exists an analytical solution to the differential equation, like Mass Spring, we use it for simulation rather than the numerical integrator. In all of the friction datasets, the friction coefficient λ is set to 0.05. The initial conditions for each dataset are sampled in the following way:

- Mass Spring - we sample q and p together from the uniform distribution over the annulus with lower radius bound 0.1 and upper radius bound 1.0 and the we multiply p by $\sqrt{km}$.
- Pendulum - we sample q and p together from the uniform distribution over the annulus with lower radius bound 1.3 and upper radius bound 2.3.
- Double Pendulum - we sample the states of both pendulums analogously to Pendulum.
- Two Body Problem - we follow the same protocol as in [11].

The exact Hamiltonians for every dataset are listed in Table 2. As mentioned in Section 4 for any of the colour datasets we randomly sample all of the hyperparameters of the system as listed. The sampling distribution for every dataset and parameter are show in Table 3.

A.1.2 Cyclic games

To produce the multi-agent cyclic games dataset, we generate ground-truth trajectories by integrating the coupled set of ODEs given by the replicator dynamics (section 4.2, equation 4) using an improved Euler scheme or RK45. In both cases the ground-truth state, i.e., joint strategy profile (joint policy), and its first order time derivative, is recorded at regular time intervals Δt. Trajectories start from uniformly sampled points on the product of the policy simplexes. No noise is added to the trajectories.

A.1.3 Molecular dynamics

In this section we provide a brief summary of the Lennard-Jones (LJ) potential and the simulation protocol employed for generating the MD datasets.

To summarise the particle interactions, let us denote all coordinates of the N-particle system by $\boldsymbol{q}^N = (\boldsymbol{q}_1, ..., \boldsymbol{q}_N)$, where $\boldsymbol{q}_i$ is the position vector of particle i. The potential energy $U(\boldsymbol{q}^N)$ can then be written as a sum over pairwise contributions,

$$U(\boldsymbol{q}^N) = \sum_{i=1}^{N} \sum_{j<i} u(|\boldsymbol{\Delta q}'_{ij}|), \tag{6}$$

where $\boldsymbol{\Delta q}_{ij} = \boldsymbol{q}_j - \boldsymbol{q}_i$ is the difference vector between particles i and j and $|\cdot|$ denotes the norm of a vector. The superscript $\cdot'$ indicates that we compute the components of the difference vector with respect to periodic boundary conditions, i.e. $\boldsymbol{\Delta q}'_{ij} = \boldsymbol{\Delta q}_{ij} - L \,\text{round}(\boldsymbol{\Delta q}_{ij}/L)$, where L is the edge length of the square simulation box and the function round is applied element-wise. The distance $|\boldsymbol{\Delta q}'_{ij}|$ corresponds to the minimal distance between the two particles on a torus. The function u models a truncated version of the spherically symmetric, pairwise LJ potential and is given by

$$u(r) = \Theta(r_\text{c} - r)\, 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right], \tag{7}$$

where Θ is the Heaviside function and r_c is a radial cutoff that is typically employed for performance reasons. The two LJ parameters ϵ and σ define the scale of energy and length, respectively. Here, we truncate interactions at $r_\text{c} = L/2$, set both LJ parameters to unity, and report all results in reduced units [59].

All simulations were performed using the simulation package LAMMPS [60]. We first initialised all coordinates and momenta randomly, followed by an energy minimisation step. We then equilibrated the system in an *NVT*-run, i.e. at a fixed number of particles N, volume V and temperature T, using a Langevin thermostat with a damping coefficient of 0.2τ, where τ is the unit of time. The thermodynamic states corresponding to both datasets are summarised in Tab. 4. The equations of motion were integrated

Particles	Temperature	Density
4	1.0	0.04
16	0.5	0.77

Table 4: Thermodynamic states at which the MD datasets were simulated. We refer to Smit and Frenkel [61] for a phase diagram of the two-dimensional LJ fluid.

using the velocity Verlet algorithm with a timestep of 0.002τ. During this $10^4\tau$ long equilibration run, we estimated the average system energy. To achieve a subsequent constant-energy (*NVE*) simulation near the target temperature in the absence of any thermostat, we followed an approach similar to the one outlined in Wirnsberger et al. [62]. To this end, we rescaled the momenta of the last frame so that its energy matched the average sampled during the *NVT*-run. After a second $10^4\tau$ long equilibration run, we then performed the $2.5 \times 10^4\tau$ long production run during which we sampled coordinates, momenta, forces and energies at constant time intervals of 0.01τ.

We repeated the above simulation procedure 100 times for each system, using different random seeds, and created the final datasets based on the combined data in a post-processing step. Further details can be found in the LAMMPS input scripts which will be made available online.

A.1.4 3D Room

For each trajectory we sample an initial radius ($r = [0.0, 0.9]$ for 3D Room Circle dataset) and angle $\theta = 0$, which we then convert into the Cartesian coordinates of the camera according to $x = r\ cos(\theta)$, $y = r\ sin(\theta)$ and $z = 1 - r$. The dynamics are created by moving the camera using step size of $1/10$ degrees in a way that keeps the camera on the unit hemisphere while facing the centre of the room. For the 3D Room Spiral dataset, the camera path traces out a golden spiral starting at the height corresponding to the originally sampled radius on the unit hemisphere, and evolving according to $r = ac^\theta$, where θ is the rotation angle measured in degrees, $c = 1.0053611$ is the spiral growth factor

Dataset	HGN	LGN	ODE	ODE[TR]	RGN Res	RGN	AR
Mass-spring	0.05(0.01)	0.01(0.00)	0.19(0.01)	0.17(0.01)	0.18(0.02)	**0.37(0.25)**	0.35(0.08)
Mass-spring +c	1.11(0.10)	**5452.37(0.01)**	1.16(0.17)	N/A(N/A)	1.52(0.40)	159.90(0.01)	123.00(15.05)
Mass-spring +c +f	1.00(0.20)	**5066.14(0.00)**	1.20(0.35)	2.11(0.68)	1.23(0.21)	5.44(4.33)	159.51(0.01)
Pendulum	1.97(0.89)	1.07(0.45)	1.44(0.31)	2.78(2.03)	2.33(1.74)	96.70(73.59)	**151.71(37.48)**
Pendulum +c	25.85(2.77)	24.66(0.97)	22.91(1.18)	29.46(5.09)	23.08(0.60)	45.93(2.20)	**240.88(25.05)**
Pendulum +c +f	14.27(0.34)	16.86(0.47)	16.05(0.47)	15.09(0.41)	15.99(0.50)	N/A(N/A)	**255.36(17.14)**
Double pendulum	27.39(3.61)	24.10(1.88)	20.52(1.19)	20.88(1.07)	19.40(0.75)	56.34(12.76)	**228.03(24.19)**
Double pendulum +c	**100.15(0.00)**	51.95(0.63)	53.66(0.69)	54.61(0.27)	58.82(0.79)	**100.15(0.00)**	**100.16(0.01)**
Double pendulum +c +f	22.57(1.18)	20.92(0.53)	26.90(3.40)	21.95(0.65)	25.55(1.53)	93.74(4.89)	**99.93(0.01)**
Two-body	0.21(0.04)	0.02(0.01)	0.33(0.09)	0.25(0.03)	0.28(0.02)	**36.73(46.84)**	0.46(0.87)
Two-body +c	10.64(4.90)	1.23(0.20)	2.20(0.13)	2.30(0.19)	1.86(0.11)	24.70(2.24)	**29.35(8.31)**
3D room - spiral	106.81(1.31)	**2778.62(0.26)**	56.33(0.42)	64.30(1.04)	47.23(0.31)	N/A(N/A)	1004.83(0.23)
3D room - circle	121.84(2.84)	193.02(0.41)	73.22(0.37)	87.37(0.71)	65.35(0.34)	N/A(N/A)	**309.47(93.14)**
MD - 4 particles	55.50(0.27)	52.26(0.33)	19.05(0.17)	28.41(0.19)	21.93(0.30)	295.19(0.05)	**342.88(10.41)**
MD - 16 particles	380.63(0.43)	350.52(0.68)	199.89(0.41)	221.97(0.93)	199.42(0.51)	**580.16(0.02)**	524.78(22.65)
Matching pennies	2.09(0.19)	1.98(0.12)	2.42(0.27)	2.22(0.21)	2.32(0.11)	3.65(0.21)	**28.07(2.80)**
Rock-paper-scissors	5.13(0.31)	4.78(0.22)	5.71(0.15)	5.77(0.32)	5.67(0.32)	**13.74(3.00)**	8.08(1.74)
Average rank	3.24	3.06	2.88	3.24	**2.71**	5.18	6.29

Table 5: Training normalized pixel mean squared error.

Dataset	HGN Forward	LGN Forward	ODE Forward	ODE[TR] Forward	RGN Res Forward	RGN Forward	AR Forward
Mass-spring	0.07(0.03)	0.02(0.00)	0.24(0.05)	0.24(0.06)	0.25(0.09)	**0.77(0.57)**	0.51(0.20)
Mass-spring +c	2.28(0.37)	**5452.37(0.00)**	2.19(0.19)	N/A(N/A)	2.81(0.96)	159.90(0.00)	209.82(12.41)
Mass-spring +c +f	2.91(0.38)	**5066.14(0.00)**	1.67(0.63)	23.00(1.76)	1.75(0.34)	52.84(26.35)	159.34(0.01)
Pendulum	27.35(8.84)	16.40(2.78)	43.11(4.84)	13.00(8.99)	50.99(17.20)	330.10(45.00)	**383.27(83.24)**
Pendulum +c	104.83(4.07)	80.73(2.24)	67.85(1.65)	76.68(8.16)	66.77(1.17)	184.12(7.06)	**390.78(14.83)**
Pendulum +c +f	106.10(1.62)	50.51(1.82)	42.24(1.49)	37.03(0.98)	44.87(1.63)	N/A(N/A)	**390.77(18.59)**
Double pendulum	175.64(1.75)	123.75(2.73)	108.71(2.59)	111.95(3.29)	105.92(2.09)	178.37(6.62)	**386.67(5.97)**
Double pendulum +c	**100.16(0.00)**	84.74(0.16)	84.45(0.21)	85.23(0.12)	86.10(0.23)	**100.15(0.00)**	**100.16(0.01)**
Double pendulum +c +f	44.54(1.00)	41.75(0.71)	44.94(2.52)	40.22(0.97)	43.70(1.26)	93.45(3.82)	**527.19(374.94)**
Two-body	18.00(3.88)	2.39(0.11)	0.82(0.46)	0.74(0.39)	0.59(0.06)	**102.51(56.70)**	3.00(5.50)
Two-body +c	49.03(7.21)	17.51(0.92)	17.12(0.47)	15.28(0.79)	15.24(0.60)	75.11(6.59)	**128.58(14.85)**
3D room - spiral	286.43(4.14)	**2786.75(0.21)**	62.11(0.82)	69.46(1.33)	57.62(0.76)	N/A(N/A)	1003.47(0.26)
3D room - circle	151.06(2.53)	198.48(0.78)	79.56(0.54)	91.04(0.55)	72.19(0.72)	N/A(N/A)	**803.05(272.54)**
MD - 4 particles	219.52(1.18)	154.30(2.15)	179.82(3.74)	134.13(1.72)	144.22(1.11)	295.19(0.05)	**500.96(12.29)**
MD - 16 particles	459.35(1.05)	481.14(1.49)	372.45(1.11)	391.89(1.61)	367.19(0.90)	580.18(0.02)	**800.18(20.80)**
Matching pennies	12.18(0.95)	10.47(0.41)	12.62(1.62)	11.57(1.02)	11.23(0.83)	23.02(5.43)	**116.65(12.64)**
Rock-paper-scissors	**223.96(14.14)**	34.70(1.40)	37.54(1.55)	37.11(2.17)	37.97(1.79)	120.44(39.30)	99.11(21.93)
Average rank	4.35	3.59	2.71	**2.29**	2.35	5.0	6.29

Table 6: Extrapolation normalized pixel mean squared error.

constant, and $a \in [0.0, 0.6]$ is the initial radius of the spiral. To generate the dataset we render the scenes into images, and use the Cartesian coordinates of the camera and its velocities estimated through finite differences as the state.

A.2 Mean squared error results

In Table 5 we report the "reconstruction" pixel mean squared error (MSE) – the most commonly used measure of model performance where the model is evaluated on how well it can reproduce the same trajectory length T as was used for training using test data. We also calculate MSE over extrapolated trajectories in Table 6, where we continue to roll out the model for a total of $2T$ steps and measure MSE over the last T timesteps. This is to check whether measuring extrapolation even over short time periods might predict the model's ability to extrapolate further in time better than the "reconstruction" MSE. In all experiments we set the value of T to 60. For more fair comparison across datasets, as proposed in Zhong et al. [23], we normalise the MSE value by the average intensity of the ground truth observation: $\text{MSE} = ||\boldsymbol{x}_t - \hat{\boldsymbol{x}}_t||_2^2 / ||\boldsymbol{x}_t||_2^2$