Learning interacting dynamical systems with latent Gaussian process ODEs Supplementary Material

Çağatay Yıldız*
University of Tübingen
cagatay.yildiz@uni-tuebingen.de

Melih Kandemir
University of Southern Denmark
kandemir@imada.sdu.dk

Barbara Rakitsch
Bosch Center for Artificial Intelligence
barbara.rakitsch@de.bosch.com

1 Appendix

1.1 Nonlinear Message Accumulations

Our formulation accumulates the messages coming from neighbors by computing their sum (Eq. **??**). Although the experimental findings indicate that our seemingly simple construction successfully learns the underlying dynamics, we introduce two straightforward extension of our framework. First, one can learn an additional non-linear function $\mathbf{f}_\text{c}$ that takes all the incoming messages as input and generates the time differential as output:

$$\begin{aligned}\frac{d}{dt}\mathbf{h}^{a_n}(t) &= \mathbf{f}_\text{c}\left(\mathbf{m}^{n1}(t), \dots, \mathbf{m}^{nN}(t)\right)\\ \mathbf{m}^{nn}(t) &= \mathbf{f}_\text{s}\left(\mathbf{h}^{a_n}(t), \mathbf{c}^{a_n}\right)\\ \mathbf{m}^{nm}(t) &= \mathbf{f}_\text{b}\left(\mathbf{h}^{a_n}(t), \mathbf{h}^{a_n'}(t), \mathbf{c}^{a_n}, \mathbf{c}^{a_m}\right),\end{aligned}$$

For notational convenience, we drop the neighboring graph from our write-up. In practice, $\mathbf{f}_\text{c}$ would only receive the messages coming from the neighbors. Since the messages interact in a non-linear way, this construction no longer disentangles the independent kinematics from the interactions.

Next, we introduce another construction in which the neighboring messages are weighted via a nonlinear function $\mathbf{f}_w$:

$$\begin{aligned}\frac{d}{dt}\mathbf{h}^{a}(t) &= \tilde{w}_{aa}\mathbf{f}_\text{s}\left(\mathbf{h}^{a}(t), \mathbf{c}^{a}\right) + \sum_{a'\in\mathcal{N}_a}\tilde{w}_{aa'}\mathbf{f}_\text{b}\left(\mathbf{h}^{a}(t), \mathbf{h}^{a'}(t), \mathbf{c}^{a}, \mathbf{c}^{a'}\right)\\ w_{aa'} &= \frac{\exp(w_{aa'})}{\exp(w_{aa}) + \sum_{n\in\mathcal{N}_a}\exp(w_{an})} \in (0,1)\\ w_{aa'} &= \mathbf{f}_w\left(\mathbf{h}^{a}(t), \mathbf{h}^{a'}(t), \mathbf{c}^{a}, \mathbf{c}^{a'}\right)\end{aligned}$$

When tested on bouncing ball datasets, this model achieved lower training and slightly higher test error (indicating overfitting). We leave further analysis of this new construction as an interesting future work.

*Work performed during internship at Bosch Center for Artificial Intelligence

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

1.2 Induced Kernel on Interaction Term

Next, we study our interaction component under a kernel perspective. In our formulation, we achieved permutation invariance across neighboring objects by aggregating all in-coming messages via the sum function (that is order invariant), similarly to what is done in standard graph neural network architectures [1].

It is interesting to see that the permutation invariance of our model formulation can also be derived from a kernel perspective. Seminal work on invariant kernels has been done by [2] and [3] who showed that a kernel is invariant under a finite set of input transformations, e.g. permutations, if the kernel is invariant when transforming its arguments. In our work, the Gaussian process prior on $\mathbf{f}_\text{b}$ induces a Gaussian process prior on the interaction term with covariance,

$$k_\text{i}(\mathbf{h}^p, \mathbf{h}^r) = \sum_{p' \in N_p} \sum_{r' \in N_r} k_\text{b}((\mathbf{h}^p, \mathbf{h}^{p'}), (\mathbf{h}^r, \mathbf{h}^{r'})),$$

which enforces the invariance by summing over all input combinations. Using a double sum is a common strategy for creating invariant kernels (see [4, 5] for a more in-depth discussion).

1.3 Experiment Details

Interaction function parameterization In all our experiments, we assume a fully connected object graph and also parameterize the interaction function $\mathbf{f}_\text{b}$ with the difference between object positions instead of absolute positions ($\mathbf{f}_\text{b}$ also takes the velocities and the global latent variables as input). Since the interactions are typically expressed in terms of distance, injecting an inductive bias of this sort helps to increase the performance as validated by our experiments.

Datasets To generate our datasets, we use the official implementations provided in [6, 7]. To ensure the accuracy of all numerical simulations in the respective code, we reduce the simulation step size. The dataset specifics are given in Table 1. Note that the noise values are proportional to the lower and upper limits of the position and velocity observations.

Table 1: Dataset details. We use the symbols P for the number of sequences, T for the sequence length, Δt for the time difference between consecutive observations, $\mathbf{s}_\text{max}$ and $\mathbf{v}_\text{max}$ for the maximum position and velocity observation, $\mathbf{s}_\text{min}$ and $\mathbf{v}_\text{min}$ for the minimum position and velocity observations, $\sigma_\mathbf{s}$ and $\sigma_\mathbf{v}$ for the standard deviation of the noise added to position and velocity observations, and T_enc for the length of the sequence needed for encoding and thereafter forward predictions.

DATASET	P_tr	P_val	P_test	T	Δt	$\mathbf{s}_\text{max}$	$\mathbf{v}_\text{max}$	$\mathbf{s}_\text{min}$	$\mathbf{v}_\text{min}$	T_enc	$\sigma_\mathbf{s}$	$\sigma_\mathbf{v}$
Noise-free bouncing balls	100	100	100	100	0.5	3.82	0.49	−3.82	−0.49	5	0.00	0.00
Low-noise bouncing balls	100	100	100	100	0.5	3.82	0.48	−3.82	−0.49	5	0.15	0.02
High-noise bouncing balls	100	100	100	100	0.5	3.82	0.48	−3.82	−0.48	5	0.30	0.04
Charges	10000	100	100	100	0.05	5.00	3.26	−5.00	−3.44	49	0.00	0.00

Dynamics approximations and hyper-parameter selection Our proposed I-GPODE method, in which the unknown independent kinematics and interaction functions are approximated with GPs, is compared with I-NODE and I-BNODE baselines in the bouncing balls experiment. To obtain the baselines, we simply replaced our GP approximation with multi-layer perceptrons (MLPs). We consider the standard, weight-space mean-field variational posterior for the BNN as done in [8]. In turn, the optimized dynamics parameters become the weights for I-NODE and the variational parameters for I-BNODE.

We perform an exhaustive comparison of hyper-parameters in different settings. In particular, for the independent kinematics function $\mathbf{f}_\text{s}$, we consider MLPs with two hidden layers and $N = 64/128/256/512$ hidden neurons, and sparse GPs with $M = 100/250/500$ inducing points. For the interaction function $\mathbf{f}_\text{b}$, we test with $N = 128/256/512$ hidden neurons and $M = 100/250/500/1000$ inducing points. We furthermore search the best activation function

among `elu/relu/softplus/tanh/swish/lip-swish` activations and consider a diagonal or full lower-diagonal approximation to the covariance matrix (of the variational posterior). We test all hyper-parameter configurations on three validation datasets with varying noise levels. We found out that the `softplus` activation and diagonal covariance approximation consistently minimize the reported metrics on validation datasets. Other hyperparameters used in our experiments are reported in Table 2. Note that the number of parameters of the simpler, non-interacting models are approximately matched with the corresponding interacting model.

Table 2: The number of hidden neurons and inducing points used in our experiments.

MODEL	$\mathbf{f}_\text{s}$	$\mathbf{f}_\text{b}$
I-GPODE	$M = 250$	$M = 250$
I-NODE	$N = 256$	$N = 512$
I-BNODE	$N = 256$	$N = 512$
GPODE	$M = 250$	-
NODE	$N = 256$	-
BNODE	$N = 256$	-

Initial value encoder Inspired by previous work [9, 10], we infer the initial position and velocity variables using a RNN-based encoder architecture. Our encoder with GRU cells processes the first five observations in backward direction: $\mathbf{Y}_5 \rightarrow \mathbf{Y}_1$. The encoder output $\mathbf{z}_1 \in \mathbb{R}^{10}$ is mapped into position and velocity initial value distributions via two separate MLPs that take the non-overlapping 5-dimensional chunks of $\mathbf{z}_1$ as input. Each MLP has the same architecture (one hidden layer, 50 neurons, ReLU activations). The model performance is somewhat robust against these encoder hyper-parameters as validated by further comparisons. We finally note that the same encoder architecture is used for GP, NN and BNN-based models.

Latent variable encoder To infer the latent variables in the charges experiment, we again utilize an RNN-based encoder. Similar to the encoder used in [7], our architecture takes the first 49 observations as input, i.e., the global latent variable $\mathbf{c}^a$ associated with object a is extracted from all available observations $\mathbf{y}_{1:49}^{1:A}$. Since the overall performance crucially depends on the hyperparameter choices unlike the initial value extraction task, we consider two sets of encoders: a "large" encoder with $\mathbf{z}_1 \in \mathbb{R}^{100}$ and an MLP with 100 neurons as well as a "small" encoder with $\mathbf{z}_1 \in \mathbb{R}^{25}$ and an MLP with 50 neurons. We furthermore perform comparisons with `relu` and `elu` activation functions for the MLP. The results in the main paper are obtained with the "small" encoder with `elu` activation, which yields the best or runner-up performance across all settings.

Training details All model variants are trained with the Adam optimizer [11] with learning rates 5e-4, 5e-4 and 1e-4 for GP, NN and BNN-based models. We perform an incremental optimization scheme with three rounds, where randomly chosen 100 subsequences of length 5, 16, and 33 are used for training. We perform 25000, 12500 and 12500 optimization iterations in each round. Training each model respectively takes 9, 3 and 12 hours on NVIDIA Tesla V100 32GB. Finally, as proposed in [8], we stabilize the BNN learning by weighting the KL term $\text{KL}[q(\mathcal{W})||p(\mathcal{W})]$ resulting from the BNN with a constant factor $\beta = D/|\mathcal{W}|$ in order to counter-balance the penalties on latent variables $\mathbf{h}^a \in \mathbb{R}^D$ and neural network weights $\mathcal{W} \in \mathbb{R}^{|\mathcal{W}|}$.

Reported metrics For a single trajectory $\mathbf{y}_{1:N}^{1:A}$, the MSE and ELL metrics are computed as follows:

$$\text{MSE} = \frac{1}{LNA} \sum_{l=1}^{L} \sum_{n=1}^{N} \sum_{a=1}^{A} \left(\mathbf{y}_n^a - \hat{\mathbf{y}}_n^{a^{(l)}} \right)^2$$

$$\text{ELL} = \frac{1}{NA} \sum_{n=1}^{N} \sum_{a=1}^{A} \log \mathcal{N}(\mathbf{y}_n^a; \hat{\boldsymbol{\mu}}_n^a, \hat{\boldsymbol{\sigma}}_n^a),$$

Table 3: A comparison of neural ODE variants on bouncing balls dataset, where the true, unknown ODE system is defined in $4D$. We suffix the dimensionality of the latent variables per object at the end of model names.

Noise Level	Model	Training		Test	
		MSE ↓	ELL ↑	MSE ↓	ELL ↑
No Noise	NODE	3.67 ± 0.35	-67.29 ± 8.04	30.07 ± 1.63	-911.95 ± 50.17
	I-NODE	6.50 ± 1.71	-36.85 ± 11.80	8.45 ± 0.93	-61.90 ± 8.63
	NODE-8	0.76 ± 0.22	-11.86 ± 9.03	29.53 ± 3.06	-885.24 ± 65.46
	I-NODE-8	2.25 ± 0.65	-14.44 ± 10.35	9.37 ± 1.45	-159.48 ± 34.13
	NODE-16	1.04 ± 0.38	-14.52 ± 6.88	26.39 ± 3.01	-615.33 ± 57.80
	I-NODE-16	1.64 ± 0.42	-4.01 ± 4.49	8.23 ± 0.81	-115.14 ± 30.98
Low Noise	NODE	2.35 ± 0.27	-34.82 ± 7.77	30.84 ± 1.94	-757.27 ± 56.12
	I-NODE	9.63 ± 0.79	-65.07 ± 11.77	13.33 ± 0.93	-96.15 ± 12.47
	NODE-8	0.76 ± 0.10	-9.12 ± 3.18	27.88 ± 1.93	-480.78 ± 46.87
	I-NODE-8	3.06 ± 1.17	-15.25 ± 4.56	15.29 ± 0.71	-167.43 ± 12.04
	NODE-16	1.46 ± 0.33	-18.71 ± 4.50	29.31 ± 2.76	-333.36 ± 30.79
	I-NODE-16	1.59 ± 0.41	-5.05 ± 2.06	15.40 ± 0.87	-155.66 ± 19.79
High Noise	NODE	2.83 ± 0.23	-43.38 ± 1.27	29.86 ± 1.38	-567.69 ± 47.49
	I-NODE	12.28 ± 1.09	-87.61 ± 13.27	15.60 ± 1.47	-128.89 ± 23.21
	NODE-8	1.14 ± 0.25	-24.70 ± 5.27	28.15 ± 1.48	-326.74 ± 32.69
	I-NODE-8	2.67 ± 0.40	-20.16 ± 2.65	20.76 ± 1.22	-154.67 ± 15.24
	NODE-16	1.94 ± 0.24	-28.39 ± 3.70	28.85 ± 3.14	-211.33 ± 21.87
	I-NODE-16	1.20 ± 0.09	-15.03 ± 1.84	20.74 ± 1.47	-146.14 ± 3.18
Higher Noise	NODE	4.02 ± 0.78	-62.19 ± 6.82	31.57 ± 0.56	-519.68 ± 37.86
	I-NODE	16.15 ± 0.21	-131.88 ± 10.11	18.35 ± 0.47	-164.60 ± 10.16
	NODE-8	1.55 ± 0.07	-44.16 ± 1.70	28.87 ± 1.09	-246.94 ± 19.09
	I-NODE-8	2.94 ± 0.42	-37.73 ± 2.34	23.54 ± 0.91	-119.53 ± 17.78
	NODE-16	2.90 ± 0.63	-44.17 ± 4.19	31.06 ± 1.21	-146.58 ± 19.16
	I-NODE-16	1.61 ± 0.22	-36.61 ± 3.79	24.67 ± 0.87	-143.61 ± 18.83

with $\hat{\mathbf{y}}_n^{a^{(l)}}$ being the l-th Monte Carlo prediction of object a at time point t_n and

$$\hat{\boldsymbol{\mu}}_n^a = \texttt{mean}\left(\{\hat{\mathbf{y}}_n^{a^{(l)}}\}_{l=1}^L\right), \qquad \hat{\boldsymbol{\sigma}}_n^a = \texttt{var}\left(\{\hat{\mathbf{y}}_n^{a^{(l)}}\}_{l=1}^L\right).$$

Finally, we report averages of the test statistics over all trajectories.

1.4 Additional Results

Latent neural ODE comparisons The ODE systems in our framework as well as the baseline models may be composed of positions, velocities and global latent variables. On the other hand, alternative black-box approaches [9, 8] typically consider a latent ODE system with arbitrary dimensionality and a VAE embedding between the observed and latent space. We compare these two modeling paradigms on the bouncing ball dataset. Since the reference methods are based on neural ODEs, we only consider neural network approximations for the differential functions. The results are presented in Table 3. In agreement with other comparisons, interaction models outperform their simpler, non-interacting counterparts. Also, latent ODE models tend to reduce the training error and increase the test error, which is a strong indicator of overfitting.

Table 4: Quantitative findings in the disentanglement task. We consider test sequences with different lengths. Note that all test sequences contain a single ball and thus the interaction function $\mathbf{f}_b$ does not play any role.

Model	$T_{\text{test}} = 10$		$T_{\text{test}} = 33$		$T_{\text{test}} = 100$	
	MSE ↓	ELL ↑	MSE ↓	ELL ↑	MSE ↓	ELL ↑
GPODE	0.21 ± 0.03	-2.37 ± 1.38	0.59 ± 0.17	-4.09 ± 3.30	9.84 ± 0.41	-17.58 ± 4.66
NODE	0.14 ± 0.05	-1.09 ± 2.51	2.90 ± 1.34	-130.33 ± 62.02	10.64 ± 2.87	-443.05 ± 145.56
BNODE	0.61 ± 0.05	-12.03 ± 1.60	2.39 ± 0.14	-23.11 ± 2.58	11.22 ± 0.54	-37.05 ± 2.95

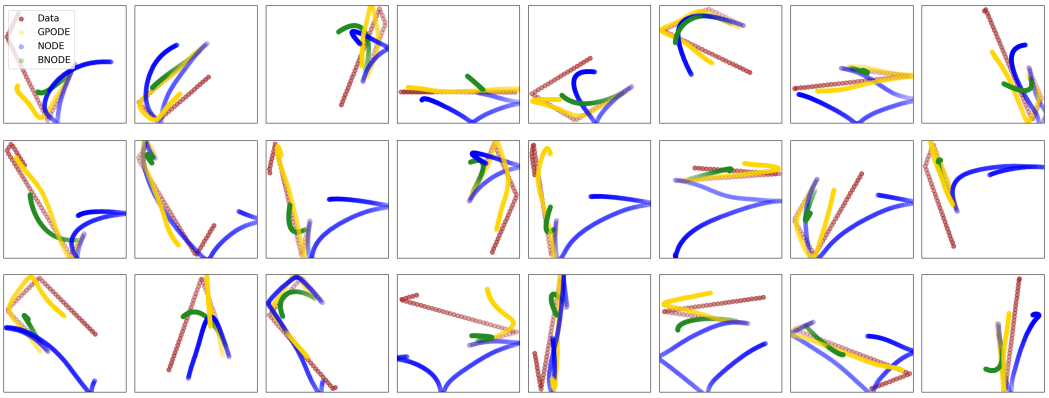

Figure 1: Additional plots comparing the independent kinematics functions of GP, NN and BNN based interacting ODE models. Above test trajectories contain single ball. Each row corresponds to one independent run of the experiment.

References

[1] Nicholas Watters, Daniel Zoran, Theophane Weber, Peter Battaglia, Razvan Pascanu, and Andrea Tacchetti. Visual interaction networks: Learning a physics simulator from video. In *Advances in Neural Information Processing Systems*, 2017.

[2] Imre Risi Kondor. *Group theoretical methods in machine learning*. Columbia University, 2008.

[3] David Ginsbourger, Xavier Bay, Olivier Roustant, and Laurent Carraro. Argumentwise invariant kernels for the approximation of invariant functions. In *Annales de la Faculté des sciences de*

Table 5: Discrete- vs continuous-time interacting GP models compared on a fully observed bouncing balls sequence with three balls.

Noise Level	Model	Training		Test	
		MSE ↓	ELL ↑	MSE ↓	ELL ↑
No Noise	I-GPODE	14.44 ± 0.69	-18.99 ± 2.12	17.29 ± 1.18	-25.88 ± 3.56
	I-GPSSM	14.80 ± 0.40	-17.16 ± 2.90	17.15 ± 1.05	-23.88 ± 4.09
Low Noise	I-GPODE	15.33 ± 0.46	-21.41 ± 4.41	17.76 ± 0.89	-26.48 ± 4.54
	I-GPSSM	14.99 ± 0.64	-17.28 ± 2.46	17.63 ± 0.79	-22.21 ± 3.51
High Noise	I-GPODE	16.10 ± 0.37	-24.05 ± 2.33	18.36 ± 0.86	-29.38 ± 1.14
	I-GPSSM	16.10 ± 0.72	-22.08 ± 2.28	18.45 ± 0.87	-24.86 ± 3.19
Higher Noise	I-GPODE	17.77 ± 1.36	-35.19 ± 1.56	20.06 ± 0.60	-41.76 ± 2.21
	I-GPSSM	17.35 ± 1.12	-33.67 ± 3.91	19.58 ± 0.45	-38.86 ± 1.68

Table 6: A comparison of NN and GP-based vanilla and interacting ODE systems on a bouncing balls dataset with two balls. All other settings are identical to the main experiment.

Noise Level	Model	Training		Test	
		MSE ↓	ELL ↑	MSE ↓	ELL ↑
No Noise	I-GPODE	10.18 ± 0.46	-6.38 ± 0.56	12.41 ± 0.90	-9.08 ± 0.71
	I-NODE	2.25 ± 0.28	-14.14 ± 4.19	4.53 ± 0.47	-56.24 ± 6.48
	I-BNODE	15.74 ± 0.63	-26.01 ± 2.23	16.33 ± 1.04	-24.18 ± 3.30
	GPODE	19.19 ± 0.72	-432.63 ± 8.55	19.68 ± 0.50	-442.38 ± 14.58
	NODE	2.99 ± 0.56	-47.59 ± 13.57	16.05 ± 0.57	-427.10 ± 32.06
	BNODE	16.73 ± 0.55	-27.50 ± 3.19	17.54 ± 0.74	-31.85 ± 1.48
Low Noise	I-GPODE	10.39 ± 0.27	-7.20 ± 0.99	13.10 ± 0.69	-10.74 ± 1.54
	I-NODE	4.39 ± 0.64	-33.49 ± 7.16	7.24 ± 0.36	-66.10 ± 3.30
	I-BNODE	16.35 ± 0.63	-25.76 ± 3.11	16.49 ± 0.51	-26.00 ± 2.96
	GPODE	20.05 ± 0.50	-450.15 ± 15.09	19.63 ± 0.55	-435.69 ± 9.70
	NODE	2.33 ± 0.53	-22.13 ± 0.17	16.17 ± 1.67	-303.51 ± 59.61
	BNODE	17.29 ± 0.52	-27.89 ± 2.60	17.88 ± 0.82	-30.06 ± 4.57
High Noise	I-GPODE	11.44 ± 0.46	-10.36 ± 1.03	13.20 ± 0.78	-13.63 ± 1.59
	I-NODE	6.13 ± 0.51	-51.98 ± 8.39	9.96 ± 0.82	-104.97 ± 9.98
	I-BNODE	15.81 ± 0.37	-27.55 ± 2.78	16.73 ± 0.59	-28.42 ± 4.55
	GPODE	19.91 ± 0.63	-423.58 ± 6.27	20.45 ± 0.31	-434.29 ± 13.16
	NODE	3.26 ± 0.64	-35.43 ± 4.23	16.99 ± 0.19	-232.94 ± 6.24
	BNODE	15.54 ± 0.22	-26.35 ± 2.56	17.01 ± 0.27	-26.08 ± 2.87
Higher Noise	I-GPODE	12.43 ± 0.55	-15.77 ± 1.66	14.83 ± 0.61	-20.28 ± 1.82
	I-NODE	8.18 ± 0.44	-72.30 ± 5.55	14.32 ± 0.95	-129.15 ± 19.09
	I-BNODE	16.16 ± 0.57	-29.08 ± 2.14	17.02 ± 0.73	-30.71 ± 3.46
	GPODE	19.68 ± 0.56	-392.48 ± 15.68	20.53 ± 0.46	-405.32 ± 12.33
	NODE	4.27 ± 0.80	-50.52 ± 2.81	18.91 ± 1.35	-228.16 ± 25.94
	BNODE	15.58 ± 0.40	-25.68 ± 2.47	16.98 ± 0.12	-27.82 ± 3.41

Toulouse: Mathématiques, 2012.

[4] David Duvenaud. *Automatic model construction with Gaussian processes*. PhD thesis, University of Cambridge, 2014.

[5] Mark Van der Wilk, Carl Edward Rasmussen, and James Hensman. Convolutional Gaussian processes. In *Advances in Neural Information Processing Systems*, 2017.

[6] Ilya Sutskever, Geoffrey Hinton, and Graham Taylor. The recurrent temporal restricted Boltzmann machine. In *Advances in Neural Information Processing Systems*, 2008.

[7] Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. In *International Conference on Machine Learning*, 2018.

[8] Cagatay Yildiz, Markus Heinonen, and Harri Lahdesmaki. ODE2VAE: Deep generative second order ODEs with Bayesian neural networks. In *Advances in Neural Information Processing Systems*, 2019.

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

Table 7: A comparison of NN and GP-based vanilla and interacting ODE systems on a bouncing balls dataset with three balls. This experiment repeats the main findings plus the training results.

Noise Level	Model	Training		Test	
		MSE ↓	ELL ↑	MSE ↓	ELL ↑
No Noise	I-GPODE	14.44 ± 0.69	-18.99 ± 2.12	17.29 ± 1.18	-25.88 ± 3.56
	I-NODE	6.50 ± 1.71	-36.85 ± 11.80	8.45 ± 0.93	-61.90 ± 8.63
	I-BNODE	20.11 ± 0.77	-36.26 ± 4.63	20.39 ± 0.62	-37.81 ± 4.23
	GPODE	22.27 ± 0.60	-553.24 ± 13.25	23.89 ± 0.91	-597.63 ± 28.48
	NODE	3.67 ± 0.35	-67.29 ± 8.04	30.07 ± 1.63	-911.95 ± 50.17
	BNODE	20.89 ± 0.97	-47.16 ± 2.66	23.22 ± 0.60	-56.57 ± 9.41
Low Noise	I-GPODE	15.33 ± 0.46	-21.41 ± 4.41	17.76 ± 0.89	-26.48 ± 4.54
	I-NODE	9.63 ± 0.79	-65.07 ± 11.77	13.33 ± 0.93	-96.15 ± 12.47
	I-BNODE	20.28 ± 1.00	-38.31 ± 7.73	20.98 ± 0.78	-36.40 ± 2.77
	GPODE	23.07 ± 0.45	-564.22 ± 16.33	23.63 ± 0.72	-580.82 ± 19.83
	NODE	2.35 ± 0.27	-34.82 ± 7.77	30.84 ± 1.94	-757.27 ± 56.12
	BNODE	20.46 ± 0.62	-45.61 ± 6.46	22.99 ± 0.77	-51.98 ± 5.68
High Noise	I-GPODE	16.10 ± 0.37	-24.05 ± 2.33	18.36 ± 0.86	-29.38 ± 1.14
	I-NODE	12.28 ± 1.09	-87.61 ± 13.27	15.60 ± 1.47	-128.89 ± 23.21
	I-BNODE	20.31 ± 0.55	-38.04 ± 2.94	20.84 ± 0.72	-47.63 ± 10.51
	GPODE	22.75 ± 1.30	-532.03 ± 21.28	24.25 ± 1.00	-569.29 ± 27.46
	NODE	2.83 ± 0.23	-43.38 ± 1.27	29.86 ± 1.38	-567.69 ± 47.49
	BNODE	19.46 ± 0.71	-43.96 ± 1.45	22.33 ± 0.52	-52.49 ± 7.41
Higher Noise	I-GPODE	17.77 ± 1.36	-35.19 ± 1.56	20.06 ± 0.60	-41.76 ± 2.21
	I-NODE	16.15 ± 0.21	-131.88 ± 10.11	18.35 ± 0.47	-164.60 ± 10.16
	I-BNODE	20.34 ± 0.52	-52.49 ± 3.60	20.88 ± 0.55	-56.44 ± 5.71
	GPODE	23.85 ± 0.88	-517.26 ± 20.19	24.71 ± 1.25	-546.95 ± 31.79
	NODE	4.02 ± 0.78	-62.19 ± 6.82	31.57 ± 0.56	-519.68 ± 37.86
	BNODE	20.07 ± 0.98	-46.07 ± 3.54	22.36 ± 0.71	-54.53 ± 2.72

Table 8: A comparison of the standard GPODE against the latent I-GPODE variants on a bouncing ball dataset without velocity observations. Note that the suffix "-S" stands for structured state space. This table repeats the findings in the main paper, plus with training results.

Noise Level	Model	Training		Test	
		MSE ↓	ELL ↑	MSE ↓	ELL ↑
No Noise	I-GPODE	23.44 ± 0.39	-299.14 ± 21.03	24.11 ± 1.18	-300.78 ± 23.35
	I-GPODE-L	22.89 ± 0.44	-167.95 ± 25.76	23.32 ± 0.76	-176.04 ± 30.98
	I-GPODE-L-S	18.09 ± 0.27	-34.69 ± 2.45	18.22 ± 1.01	-33.45 ± 1.96
Low Noise	I-GPODE	23.50 ± 0.40	-243.65 ± 9.69	23.26 ± 0.77	-239.37 ± 11.10
	I-GPODE-L	49.26 ± 51.33	-810.91 ± 1025.94	47.01 ± 47.38	-760.60 ± 921.56
	I-GPODE-L-S	20.64 ± 0.21	-78.32 ± 22.79	20.46 ± 0.92	-76.35 ± 22.48
High Noise	I-GPODE	23.42 ± 0.89	-241.66 ± 17.19	23.77 ± 0.68	-254.34 ± 16.43
	I-GPODE-L	119.49 ± 96.91	-1306.78 ± 1277.32	119.24 ± 94.27	-1320.27 ± 1282.10
	I-GPODE-L-S	21.51 ± 0.45	-116.33 ± 11.57	21.80 ± 0.69	-115.50 ± 19.20
Higher Noise	I-GPODE	24.06 ± 0.38	-271.81 ± 9.72	24.57 ± 1.27	-267.08 ± 26.09
	I-GPODE-L	78.00 ± 29.98	-448.65 ± 299.58	78.49 ± 30.54	-450.42 ± 301.84
	I-GPODE-L-S	23.27 ± 0.49	-141.95 ± 16.96	23.64 ± 1.38	-147.35 ± 21.42

Table 9: A comparison of NN and GP-based interacting ODE systems on the charges dataset. Note that here we repeat the main findings plus the training results.

	Model	Training		Test	
		MSE ↓	ELL ↑	MSE ↓	ELL ↑
Without Global Latents	I-GPODE	19.3 ± 0.5	-170 ± 6	19.2 ± 0.9	-171 ± 7
	I-NODE	14.5 ± 0.7	-471 ± 23	14.1 ± 0.8	-460 ± 36
With Global Latents	I-GPODE	15.4 ± 1.8	-171 ± 85.7	15.4 ± 2.3	-172 ± 88
	I-NODE	9.8 ± 0.4	-271 ± 12.3	9.9 ± 0.6	-282 ± 14
	I-GPODE-S	12.8 ± 1.7	-176 ± 61	12.9 ± 2.1	-177 ± 60
	I-NODE-S	9.6 ± 0.3	-271 ± 12	9.7 ± 0.2	-282 ± 8
Globals Observed	I-GPODE	10.6 ± 0.6	-94 ± 8	10.7 ± 1.1	-97 ± 9
	I-NODE	7.3 ± 0.8	-140 ± 21	7.5 ± 1.2	-148 ± 27

[10] Pashupati Hegde, Çağatay Yıldız, Harri Lähdesmäki, Samuel Kaski, and Markus Heinonen. Bayesian inference of ODEs with Gaussian processes. *arXiv preprint arXiv:2106.10905*, 2021.

[11] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

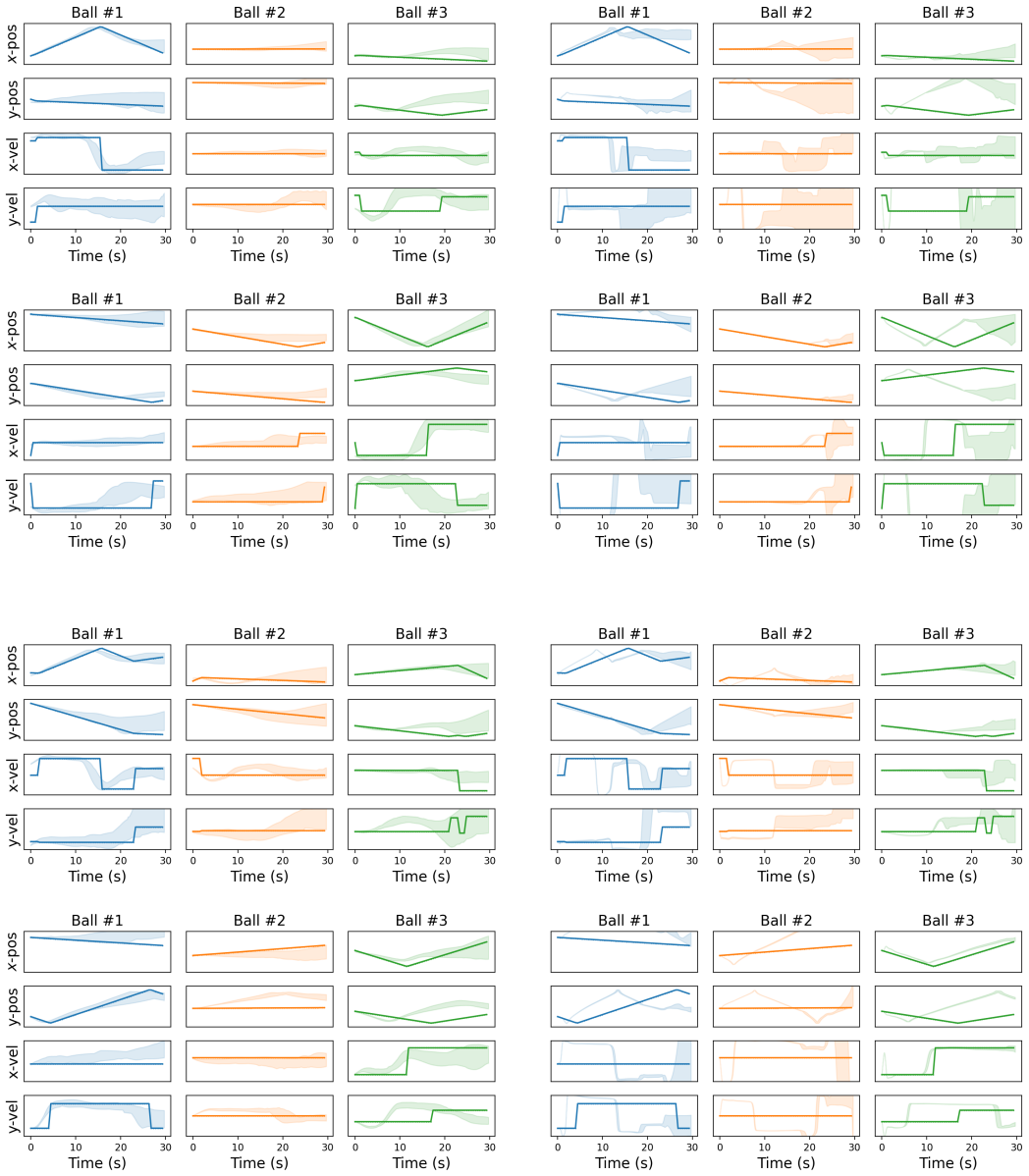

Figure 2: Additional plots comparing the uncertainty quantification of I-GPODE (left) and I-NODE (right) on bouncing balls dataset. The first two row groups show the training fits and the last two groups are the test predictions.