Uncertainty Quantification using Deep Ensembles for Decision Making in Cyber-Physical-Human Systems

Muhammad Bilal Shahid*
Iowa State University, Ames, IA 50011

Robert Robison†, Tom Shafer‡, and Victor Diloreto§
Elder Research, Inc, Charlottesville, VA 22903

Natalia M. Alexandrov¶
NASA Langley Research Center, Hampton, VA 23681

Cody Fleming‖
Iowa State University, Ames, IA 50011

In this paper and its companion, *Differential Equation Approximation Using Gradient-Boosted Quantile Regression*, Robison et al., we examine an approach to quantifying model uncertainty with the aim of increasing the trustworthiness of computational models in human-machine interactions. In *Differential Equation Approximation Using Gradient-Boosted Quantile Regression*, we focus on gradient-boosted decision trees, while in this one, we give more details about deep ensembles.

Uncertainty quantification is crucial for building trustworthy autonomous decision-making agents in human-machine teams. There are two types of uncertainties: *aleatoric* and *epistemic*. The former is related to the inherent stochasticity (noise) of the process, whereas the latter is associated with the lack of knowledge or representation capability of models, such as neural networks. By lack of knowledge, we mean the model's inability to accurately predict outputs for all possible inputs. The aleatory uncertainty can be estimated fairly easily with, for example, filters, whereas epistemic uncertainty is challenging to compute. This paper uses deep ensembles to quantify both aleatory and epistemic uncertainty. It can act as an uncertainty-aware surrogate transition model for decision-making frameworks. "Uncertainty-aware" means that the surrogate transition model should make predictions along with confidence in those predictions. In the context of decision-making, the transition models are ordinary differential equations (ODEs). Since ODEs can be simulated to make one-step or multi-step predictions, a good surrogate model for them should perform reasonably well in both modes. In a multi-step approach, the trajectory sampling method TS∞ was used to propagate uncertainty over multiple steps. The cartpole dynamical system was selected to demonstrate the ability of deep ensembles as good surrogate transition models for decision-making frameworks. The deep ensembles modeled the dynamics of cartpole ODEs and made uncertainty-aware predictions in single-step and multi-step transition modes.

I. Introduction

Most systems are stochastic by nature and can be uncertain due to many factors, such as environmental changes, faults in components, inherent variability, etc. This property of systems can make their behavior challenging to predict and, hence, difficult to control. To predict the behavior of such systems, we would ideally like to obtain exact transition models, but this is almost always infeasible in practice. Statistical techniques have historically been used to overcome

*Graduate Research Assistant, Mechanical Engineering.
†Senior Data Scientist.
‡Principal Data Scientist.
§Chief Technology Officer.
¶Senior Research Scientist, AIAA Associate Fellow.
‖Associate Professor, Mechanical Engineering, AIAA Associate Fellow.

this challenge, and recent advances in machine learning and artificial intelligence have provided even more powerful and expressive ways of modeling dynamic, stochastic systems.

These approaches result in an approximation of the true model that can be learned based on data collected from the system. Because it is an approximate model, it can sometimes be inaccurate for the purposes of the context of operation, which is especially detrimental for mission- or safety-critical systems. Hence, we need learned models that can not only predict the system's behavior but also express their confidence in those predictions. That is, we need models that can inform system operators and other stakeholders that it is unsure of its predictions. This is termed uncertainty quantification (UQ) and propagation and is an essential property of a predictive model in many applications, such as decision-making and control, diagnostics, fault detection, etc. The predictive models used here are neural networks whose generalization properties are not well understood [1]. By ***generalization***, we mean the performance of a trained neural network on data that did not participate in the training process. Hence, quantifying uncertainty for neural networks is of prime importance. It shows that generalization and UQ are connected topics as the need for UQ arises because the neural networks do not generalize well or at all to off-training data.

The generalization of neural networks is a complicated topic [2] and plays a pivotal role in understanding neural networks [1]. The assumption of IID (independent and identically distributed) is prevalent in deep learning (and learning theory in general) [3], which assumes that the data points are independent and come from the same distribution. It is used to quantitatively characterize the generalization performance of neural networks on unseen InD (in-distribution) data via generalization bounds [4]. These bounds aim to demystify the reason behind the generalization gap: the performance difference between a trained neural network on training data and unseen InD data [5] *. However, predicting the performance of neural networks on the OOD (out-of-distribution) data is challenging [6]. They tend to generate overconfident predictions and can be calibrated for improved generalization [7]. Wald et al. drew a link between calibration and OOD generalization and introduced multi-domain calibration to improve OOD dataset performance [8].

Let us consider the term "UQ" in more detail. According to IID assumption, the data points are treated as independent samples from the same distribution which is generally assumed to be Gaussian, and the objective of a neural network is to learn the parameters of that distribution. In the literature on UQ, this is known as ***aleatory*** uncertainty, which is related to a process's inherent stochasticity (noise). Another notion of uncertainty is ***epistemic*** uncertainty, which is related to the lack of knowledge about the underlying process. It quantifies the black-box model's lack of knowledge about the process it was trained to simulate. By ***lack of knowledge***, we mean the model's inability to make accurate predictions for all possible inputs. The net uncertainty of a stochastic process modeled with a black-box (such as neural networks) adds to both types of uncertainties; that is, aleatory and epistemic. The former is easy to learn with filters, Gaussian processes (GPs), or a neural network trained with Gaussian loss. The latter is very difficult to estimate accurately. Even though several black-box models can learn aleatory uncertainty, they fail to estimate epistemic uncertainty [9].

Deep Ensembles have emerged as a promising method for UQ in deep learning [10–12]. A deep ensemble uses the notion of bootstrapping, creating a collection of neural networks that independently discover different minima in highly non-convex loss spaces, due to the parameter optimization process initiation from different locations in parameter space. The same query can be input to neural networks in deep ensembles at inference time, each giving their respective mean (prediction) and variance (confidence in prediction). The means and variances of the ensembles can be aggregated to generate the aggregated mean and variance.

A cartpole system with various disturbances and sources of stochasticity serves as the basis for this study. The deep ensembles will serve as surrogate models for cartpole dynamics. After training, they can be tested for one-step or multi-step predictions [†]. Both testing modes are popular and mainly depend on the problem under consideration. Testing the model in multi-step or free simulation mode is considered a robust testing approach as uncertainty increases under longer prediction horizons, making accurate predictions increasingly challenging. During testing, the aggregated mean can be compared with the ground truth. The deep ensembles are uncertainty-aware if the aggregated mean is far from the ground truth and the aggregated variance is high. On the other hand, if the aggregated variance is small in the same scenario, the deep ensembles are overconfident, resulting in wrong prediction and inaccurate uncertainty quantification. It is challenging to say confidently that the deep ensembles will always make conservative uncertainty-aware predictions for all possible inputs. *Nonetheless, they will be tested on diverse data with different control policies and noise levels to ensure that they are good at letting us know when they are wrong*. This study aims to test the performance of deep ensembles with these criteria.

*An example of unseen InD data is validation data that are kept aside before training the network to benchmark the performance.

[†]In single-step predictions, the model sees input from the ground truth set at each time step, whereas in multi-step predictions, the model sees its own prediction from the last time step. The latter transition mode accumulates errors over time since the model's predictions are erroneous.

II. Background

Various approaches have been explored to test the performance of uncertainty-aware surrogate models for cartpole dynamics. They can be broadly classified into two types: Bayesian and ensemble-based. The Gaussian process (GP) belongs to the former and deep ensembles belong to the latter. Even though deep ensembles arguably belong to both types, they will be classified as an ensemble method here. The deep ensemble as a Bayesian method will be discussed later. The UQ in deep learning is an active topic, and the literature review in this section is not comprehensive. Here, we discuss the UQ methods that we tested on the cartpole problem. For a detailed review of UQ, the reader is referred to Abdar et al. [13].

One line of work in UQ focuses on GP, especially in Bayesian optimization, where GP is used to form surrogate models of objective functions [14]. The GP computes uncertainty estimates for the function being modelled [15], by construction. Examples of kernels used as similarity metrics in GP are squared exponential and rational quadratic. These kernels are interpolators that lack the expressive power of neural networks and perform poorly in pattern discovery and extrapolation [16]. In addition to extrapolation, some variants of GP, e.g., the exact GP [‡] suffer from scalability issues due to the large kernel matrix; although there have been efforts to improve scalability [18]. The large kernel matrix can be approximated with a smaller matrix, using inducing point methods, where the inducing points that form the smaller matrix can be learned via variational learning [19]. Hoffman et al. introduced stochastic variational inference approach for scalable approximation of posterior distributions with stochastic optimization methods, such as stochastic gradient descent [20]. There are many efforts toward improving scalability and speed of inference with GP. Several are implemented as a part of the GPyTorch library [17]. GPyTorch improves GP scalability with GPU acceleration. However, their use is not widespread in the realm of epistemic uncertainty quantification, given GP's inability to capture different modes of the posterior distribution [9].

Another line of UQ work focuses on ensemble methods, where the focus is on increasing the *predictive diversity* of black-box models in an ensemble. In one of the recent works in this line, by Lakshminarayanan et al. [10], the neural networks in an ensemble were trained with negative log-likelihood (NLL) as a loss function. This gives a neural network the ability to predict both the mean and the variance for a given input. This method exhibited state-of-the-art performance in quantifying uncertainty for deep learning at the time of publication.

Since then, several works have been published that indicated higher predictive diversity than basic deep ensembles. Jain et al. used Maximized Overall Diversity (MOD) loss to achieve improved uncertainty estimates and diverse predictions [21]. The vanilla-style deep ensembles use the same architecture for all networks in an ensemble, whereas using different architectures for all networks in an ensemble improves predictive uncertainty and diversity [22].

It is well known that the domains of UQ and generalization overlap. The neural tangent kernel (NTK) was introduced to study the convergence and generalization of neural networks. Bayesian deep ensembles via NTK were developed, resulting in improved uncertainty estimation by giving conservative prediction compared to vanilla-style deep ensembles [23]. Another approach used SGD with warm restarts, a learning rate scheduler, to converge the same neural network to different minima along its training path. The weight parameters corresponding to different minima can be considered as one ensemble [24]. The resulting ensemble did not outperform vanilla-style deep ensembles, but it is computationally cheaper.

These and many other methods that have sought improvement over vanilla-style deep ensembles do so by promoting *predictive diversity* among outputs of neural networks in an ensemble. Even though these methods achieve a minor improvement in terms of predictive diversity over vanilla-style deep ensembles, they do so at a higher computational cost, with the exception of Huang et al. [24], whose method was computationally cheaper but did not outperform basic algorithms. We used vanilla-style deep ensembles because there is no unequivocal gain in using the alternatives.

III. Understanding Deep Ensembles

The performance of deep ensembles was not well understood initially, but several papers have since attempted to demystify the behavior of deep ensembles by studying the loss landscape of neural networks [25] or making connections with Bayesian Deep Learning [9].

The loss landscape of neural networks is highly non-convex [26], has multiple minima, and convergence to a minimum depends on the initial values of weight parameters of the neural network and the stochastic optimization trajectory it takes to reach the minima [25]. A graphical illustration in Figure 1 demonstrates this property. In the figure, two randomly initialized neural networks start from *Init 1* and *Init 2* and converge to their respective optima *Optimum 1*

[‡]The term "exact" is used to differentiate exact GP from other variants that approximate a large kernel matrix with smaller matrices and exploit the structure of matrices to yield fast matrix-vector multiplication for faster inference [17],

and *Optimum 2*. A low-loss tunnel can connect these two optima, shown as a green dotted line along which the loss remains almost constant, while we traverse from one mode (*Optimum 1*) to another (*Optimum 2*). The so-called mode connectivity has inspired a new UQ approach known as Fast Geometric Ensembling [27].

The ability of deep ensembles to discover various modes of loss space is key to its better performance over competing approximate Bayesian inference approaches, such as Variational Bayesian [28], Monte Carlo Dropout [29], MultiSWAG [9], among others. The black line in Figure 1 is a path constructed by linear interpolation to connect two modes. Its significance is in added understanding of the optimization in high-dimensional spaces [30].

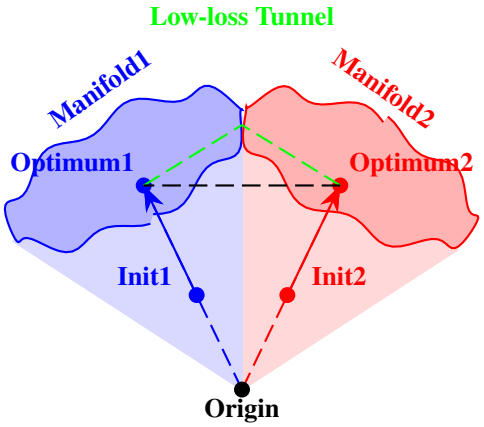

Fig. 1 A graphical illustration for two minima corresponding to two randomly initialized SGD trajectories, namely Init 1 and Init 2

The second perspective on deep ensembles is through the lens of Bayesian deep learning, as explained by Wilson et al. in [9] and an associated blog post [31], which takes the view of deep ensembles as one of the many methods of approximate Bayesian inference [32]. In Bayesian learning, the predictive distribution is given as:

$$p(y|x,\mathcal{D}) = \int p(y|x,w)p(w|\mathcal{D})dw \tag{1}$$

The predictive distribution of interest ($p(y|x,\mathcal{D})$) [§] is difficult to calculate because it requires numerical integration of the high dimensional integral in Eq. 1. The operation is also known as the marginalization of parameters (w). For neural networks, the integral's dimension can vary from thousands to billions, depending on the number of weight parameters. The deep ensembles approximate this integral with a small number of neural networks, and the resulting approximation has been shown to give reasonable predictive uncertainty for most applications. In that sense, it is reasonable to consider the approach as an approximate Bayesian inference method, as argued by Wilson et al. in [31].

The Gaussian processes (GP) and approximate Bayesian inference methods do not work well because they give a Gaussian approximation to this otherwise highly multi-modal integral [9, 25]. A graphical illustration in Figure 2 shows that Variational Bayesian captures only one mode very well, whereas deep ensembles capture multiple modes. However, they do not fully capture the uncertainty within each mode. This issue was addressed by Wilson et al. in [9] where each mode was modeled with low-dimensional Gaussian during training, to sample many neural networks at inference time, thus better capturing the uncertainty of each mode.

IV. Problem Description

To demonstrate the effectiveness and potential of deep ensembles in quantifying uncertainty in dynamic systems, the cartpole problem is used throughout the rest of the paper. A cartpole's state can be described with four variables: $\theta, \dot{\theta}, x, \dot{x}$. These variables characterize the motion of a cartpole and can be controlled with a horizontal force (f) that acts as a control input. The state variables and control input can be seen on the physical model of a cartpole shown in Figure 3. The equations of motion are derived based on the physical model and are given below. For detailed derivation, the reader is referred to Cannon et al. [33].

[§]The x in 1 denotes the input feature vector whereas the x in section IV denotes the distance of the center of cart from the y-axis.

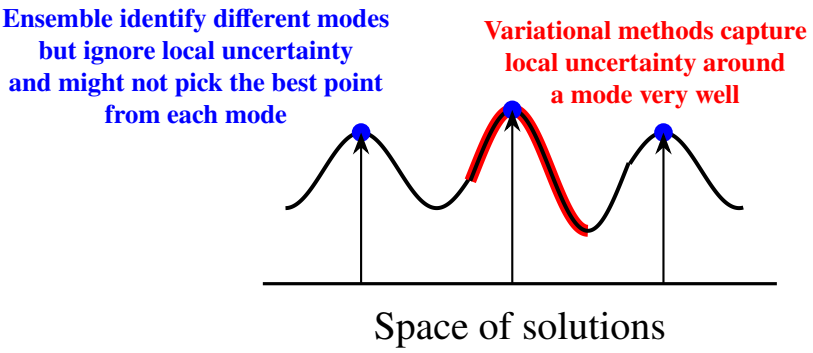

Fig. 2 A graphical illustration to compare Variational Bayesian and Deep Ensembles

$$\ddot{x} = \frac{f - ml\ddot{\theta}\cos\theta + ml\dot{\theta}^2\sin\theta + F_f}{M} \tag{2}$$

$$\ddot{\theta} = \frac{Mg\sin\theta - \cos\theta\left(f + ml\dot{\theta}^2\sin\theta + F_f\right) - \frac{MM_f}{ml}}{ml\cos^2\theta - (1+k)Ml} \tag{3}$$

where

m = Mass of the pole (kg)
m_c = Mass of the cart (kg)
M = Combined mass of the pole and the cart (i.e., M = m+m$_c$)
l = Length of the pole (m)
g = Gravitational constant (m/s^2)
f = An input force applied to the cart to push it to the left or right (in Newtons). A positive value means a push to the right and a negative value means a push to the left.
x = Horizontal position of the cart on the track as measured from the origin (m)
$\dot{x}, \ddot{x}$ = The velocity and acceleration of the cart, respectively
θ = Angle of the pole (rad)
$\dot{\theta}, \ddot{\theta}$ = Angular velocity and acceleration of the pole, respectively (rad/s)

The rotational inertia of a point mass (m) rotating at a fixed distance l from the axis of rotation is $J = ml^2$. Cannon et al. [33] use $J = ml^2/3$, which is the moment of inertia for a rod whose mass is uniformly distributed along its length. Other definitions of J are multiples of ml^2; hence, k is used to express the moment of inertia in a general form, and an appropriate value of k should be substituted for simulation. In the present case, $\frac{1}{3}$ was used in place of k during simulation, to follow the convention of Cannon.

The two friction terms, F_f and M_f, appearing in the above equations express the cart-track and cartpole frictions, respectively. Following the convention of Cannon [33], F_f was considered proportional to the velocity ($\dot{x}$) of the cart (i.e., $F_f = -\mu_c\dot{x}$) and M_f to the angular velocity ($\dot{\theta}$) of the pole (i.e., $M_f = \mu_p\dot{\theta}$). It is worth noting that many complex friction models can account for the cart-track and the cartpole friction in the dynamic model; however, simpler ones were used to avoid any discontinuities.

V. Data Collection

The dynamic model described in IV was used to generate the dataset for the data-driven uncertainty quantification (UQ). Physical parameters m, m_c, l, k, μ_c, and μ_p must be specified for the model's use. The following values of the parameters were chosen arbitrarily: $m = 0.1$ kg, $m_c = 1$ kg, $l = 0.5$ m, $k = \frac{1}{3}$, $\mu_c = 0.0005$, and $\mu_p = 0.000002$.

We used three trajectories in this paper: training, validation, and test. The initial conditions for all three are shown in Table 1. The validation trajectory is an extension of the training trajectory, while the test trajectory is completely

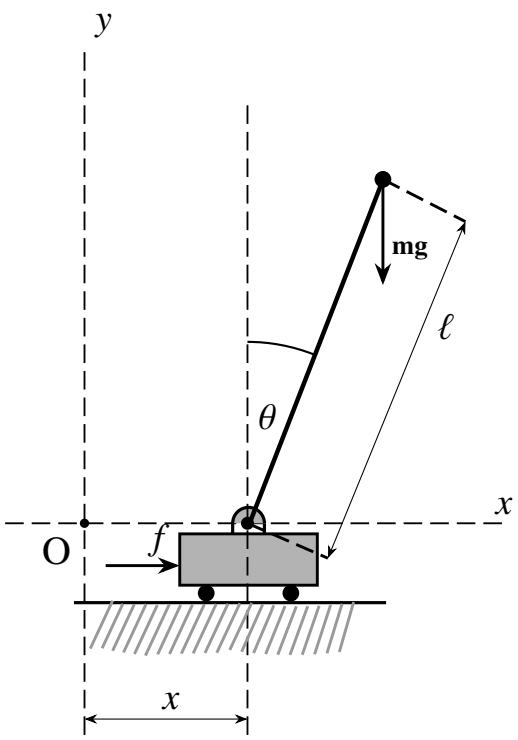

Fig. 3 A cartpole dynamical model

Table 1 Data Collection

Variable	Train	Validation	Test
θ	0.3	209.4	-0.1
$\frac{d\theta}{dt}$	1.0	-3.4	-3.0
x	0.0	-41.0	0.0
$\frac{dx}{dt}$	-0.72	0.37	0.15
Control Policy	$F(t) = 5\sin(6t)$	$F(t) = 5\sin(6t)$	$F(t) = -4\cos(4.5t)$
Seconds	500	100	100

new, with a different control policy. The control policies were chosen to enable a thorough exploration of the feasible parameter space.

We used the Runge-Kutta 45 scheme to solve the Initial Value Problem (IVP) for each trajectory, saving the data at a time step of 0.01 seconds. Our initial goal was to evaluate on all 100 seconds of the validation and test data, but this was changed to the first 10 seconds, since all tested methods had prediction intervals that diverged before that time. Finally, we generate the data three times–once with no noise, once with 0.1 scale Gaussian noise, and once with 0.5 scale Gaussian noise. The code is available online [¶].

VI. Methodology

Given a training dataset, deep ensembles can learn the dynamics of a system in a black-box manner. To that end, the dataset was collected as described in section V—the input features are chosen as state variables (s = $[\theta_t, \dot{\theta}_t, x_t, \dot{x}_t]$)

[¶] `https://github.com/coordinated-systems-lab/UncQuanNASA`. The codebase contains several algorithms that were used for the approximation of cartpole dynamics and UQ. For instance, SINDy [34] was used to extract the closed-form approximation of the dynamics but failed to do so. The GP were used for approximation of dynamics and UQ but failed to do well on the latter.

and control input ($a = [f_t]$) [‖] whereas outputs are the differences between the state variables at the next time step and the current time step ($\Delta\theta$, $\Delta\dot{\theta}$, Δx, and $\Delta\dot{x}$). To further elaborate on the output of deep ensemble, $\Delta\theta$ can be written as $\theta_{t+1} - \theta_t$. It is worth mentioning that the conversion of theta in the interval $[0, 2\pi]$ positively impacted the predictive performance. Hence, the feature θ was converted to the specified range only for inputting it into the model, but the prediction is still the difference of θ in its original form. There is a certain merit to predicting all four features as outputs with deep ensembles instead of $\ddot{\theta}$ and $\ddot{x}$. Since deep ensembles give predictive distributions as outputs (aleatory and epistemic uncertainty) instead of point estimates, we can have predictive distributions directly over our outputs of interest (i.e., θ_{t+1}, $\dot{\theta}_{t+1}$, x_{t+1}, $\dot{x}_{t+1}$). We can get outputs of interest by adding the input states (θ_t, $\dot{\theta}_t$, x_t, $\dot{x}_t$) and the outputs ($\Delta\theta$, $\Delta\dot{\theta}$, Δx, and $\Delta\dot{x}$). This is known as forward dynamics. It has been shown to perform well in deep reinforcement learning (RL) [35] and controls frameworks with black-box transition models [36].

As the name suggests, a deep ensemble is an ensemble of neural networks trained together using Gaussian loss whose mathematical form is shown in Eq. 4. The neural network outputs are the mean $\mu(s, a)$ and the variance $\sigma(s, a)$ of the target distribution for a given state-action pair (s, a). The subscripts i, $|D|$, and y in Eq. 4 denote the example number, number of training examples (or examples in a mini-batch), and target output.

$$\mathcal{L}_D = \frac{1}{2}\sum_{i=0}^{|D|}\left[\log\ \sigma(s_i, a_i)^2 + \frac{(y_i - \mu(s_i, a_i))^2}{\sigma(s_i, a_i)^2}\right] \tag{4}$$

The deep ensembles excel at estimating both aleatory and epistemic uncertainties. The aleatory uncertainty comes from Gaussian loss and epistemic comes from ensembling (or aggregating) the outputs of neural networks in the deep ensembles. The performance of deep ensembles in this context is well documented in the RL literature [37]. Since there are many neural networks in deep ensembles, the aggregation method of their respective outputs is very important for estimating correct uncertainty. The final output $\mu(s, a)$ can be randomly selected out of N outputs, where N denotes the total number of neural networks in a deep ensemble. The reason for random selection can be understood in terms of non-convexity of the loss space of neural networks. There are multiple minima in the non-convex loss space, and selecting the right minimum for a given state-action pair (s, a) is a challenging problem. So, one of the minima can be randomly trusted with the aggregated variance. This also indicates the importance of aggregated variance. The current work uses the *ensemble variance* as a variance aggregation method. It can be written mathematically as follows:

$$\sum\nolimits^{*}(s, a) = \frac{1}{N}\sum_{i}^{N}((\sum_{\phi}^{i}(s, a))^2 + (\mu_{\phi}^{i}(s, a))^2) - (\mu^*(s, a))^2 \tag{5}$$

$$\mu^*(s, a) = \frac{1}{N}\sum_{i}^{N}(\mu_{\phi}^{i}(s, a)) \tag{6}$$

Where $\sum^*$ denotes the aggregated variance. There are several methods of aggregating variances of neural networks in deep ensembles, but ensemble variance has been shown to correlate well with the mean-squared error (MSE) of the transitions. Ideally, the aggregated variance should be directly proportional to MSE, but this direct relation is hard to achieve. The ensemble variance best fulfills this criterion amongst other variance aggregation methods [37].

VII. Results

As discussed, the performance of deep ensembles to make uncertainty-aware predictions will be tested in single-step and multi-step transition modes. The performance of deep ensembles in both modes will be discussed in this section.

A. Single-step Performance

As described earlier in section V, three noise levels were injected into the data. The results for the data with high noise (zero mean and 0.5 variance) are given in this section for the train, test, and validation datasets in Figure 4 (a), (b), and (c), respectively. The results for low noise (zero mean and 0.1 variance) and deterministic data (no noise) are shown in Figs. A.1 and A.2, respectively.

[‖] The **s** and **a** are used to denote the state and action vectors, respectively. This is in accordance with the RL literature.

The one-step predictions are accurate for all features. For features x and $\dot{x}$ in Figure 4, the effect of noise is more pronounced compared to θ and $\dot{\theta}$, and the model is showing that noise by raising its confidence bounds shown in blue shades. These confidence bounds correlate well with the noise artificially injected into the data. This shows that deep ensembles can learn noise (aleatory uncertainty).

In real systems, the sensors produce noisy observations, and the ability to predict the next state of the system from those observations is a crucial feature of a good predictive model. The stochastic transition models are very useful for decision-making algorithms, such as RL, where the algorithm predicts an optimal action based on the next state predicted by the surrogate transition model.

As discussed in section VI, the net uncertainty for a stochastic process is the sum of aleatory and epistemic uncertainties. However, Figure 4 manifests *mainly* aleatory uncertainty, which was artificially injected into the features to assess the performance of deep ensembles. To support this claim, we observe that the noise is not as pronounced in A.1 (low noise) and vanishes almost completely in A.2 (no noise). This shows that the uncertainty in Figure 4 is indeed aleatory. We do not observe epistemic uncertainty because the neural networks in deep ensembles are confident about the accuracy of their predictions. The epistemic uncertainty will appear when the neural networks in deep ensembles show a disagreement in their respective predictions, which is not the case here.

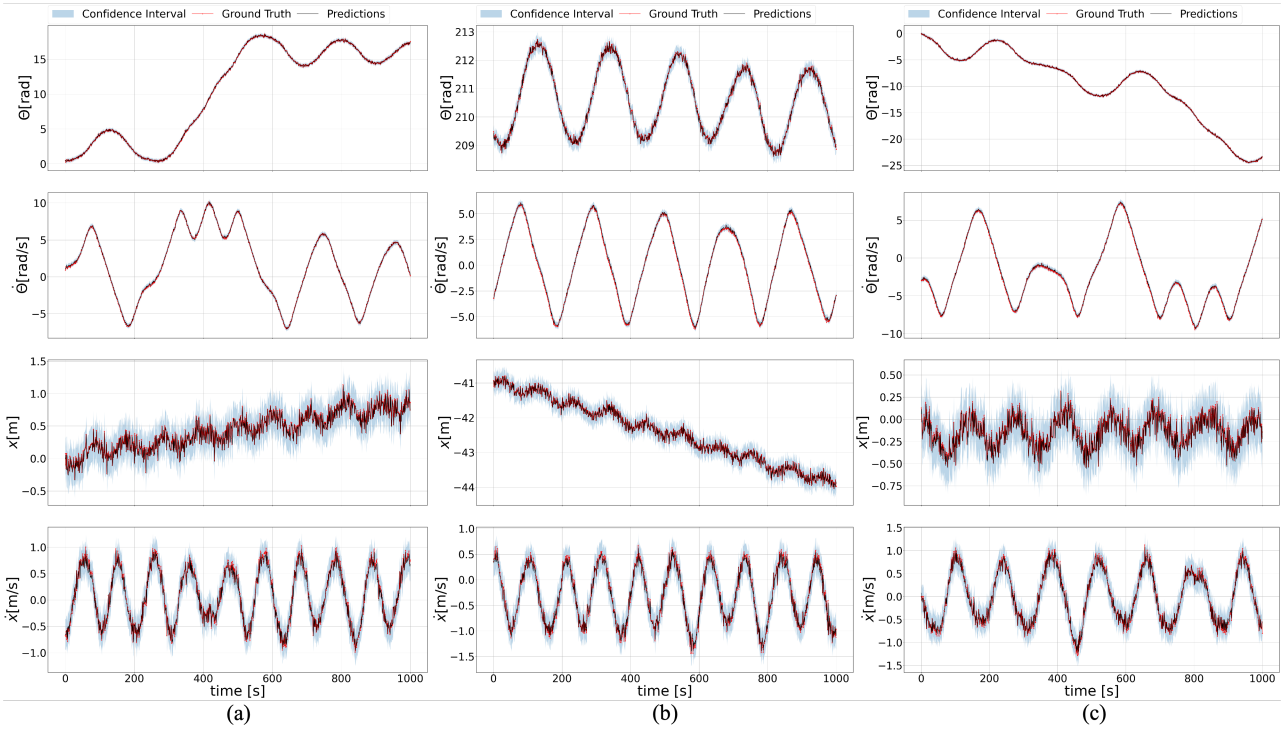

Fig. 4 Single-step performance of deep ensembles on high noise data (a) Train (b) Validation (c) Test

B. Multi-step Performance

If a transition model can predict further into the future, decision-making algorithms can make more informed decisions. One such example is model predictive control (MPC), where the action at the current time-step is issued by looking a number of steps ahead into the future. However, predicting the system state a number of steps ahead is not trivial for a transition model: errors accumulate when predictions at time-step (t) are fed to the transition model to compute predictions at time-step ($t+1$) multiple times over the prediction horizon. The term "prediction horizon" denotes the number of look-ahead steps needed to optimize the action at the current step.

If the transition model is stochastic and predicts a distribution over the next state of the system rather than a point estimate, it becomes even harder to deal with the multi-step predictions, since we now need to propagate uncertainty into the future. Deep ensembles are stochastic transition models; hence, uncertainty must be propagated into the future to get robust uncertainty estimation for future time steps.

To propagate uncertainty, we use the trajectory sampling (TS) method from Chua et al. [38], where particles P are sampled from neural networks in deep ensembles. If B is the number of neural networks in an ensemble, each network b propagates $\frac{P}{B}$ number of particles at any time step over the prediction horizon. The particle p is propagated according to $s_p^{t+1} \sim f_{\theta_{b(p,t)}}(s_p^t, a^t)$.

There are two trajectory sampling methods: **TS1** and **TS**∞. The latter performs better for MPC applications and is used in the current work. In **TS**∞, each particle's bootstrap index remains constant throughout the propagation. This captures the time-invariance assumption of the dynamics function's (f) uncertainty. To clarify this step, consider neural networks in deep ensembles as representing the uncertainty in the function space of the true dynamic function (f). which we assume to be time-invariant.

In terms of Bayesian deep learning [9], the posterior distribution represented by neural networks is the same at each time step, and the neural networks in the deep ensemble can be thought of as posterior samples that remain unchanged over the prediction horizon. One advantage of **TS**∞ is that the uncertainty can be broken down into its two parts: aleatory and epistemic [39], which is useful for algorithms that perform exploration based on uncertainty. An exploration algorithm, such as Bayesian optimization, should not explore based on aleatory uncertainty which is inherent in the process. The purpose of exploration is to explore unexplored parts of the function space - the domain of epistemic uncertainty.

The multi-step predictive performance of deep ensembles for high noise is shown in Figure 5 for the training, validation, and test datasets. The predictive performances for low noise and deterministic cases are shown in Figures A.3 and A.4, respectively. Figure 5 shows that the model can predict accurately in the first few time steps; hence, the uncertainty is small. In the later time steps, the model cannot predict accurately, therefore, the uncertainty is very high, as indicated by the blue-shaded confidence bounds.

Unlike in the single-step case, the model shows both aleatory and epistemic uncertainty. It is worth comparing the scale of the y-axis for different levels of noise (Figs. 5, A.3, and A.4). The model's uncertainty is highest in Figure 5, where the high level of noise was injected. This happens because the state to be propagated to the next time step is not always the mean. Since neural networks in deep ensembles predict both the mean and the variance, a state is sampled from that distribution for propagation to the next time step for any particle in trajectory sampling. This highlights the importance of modeling aleatory uncertainty along with epistemic.

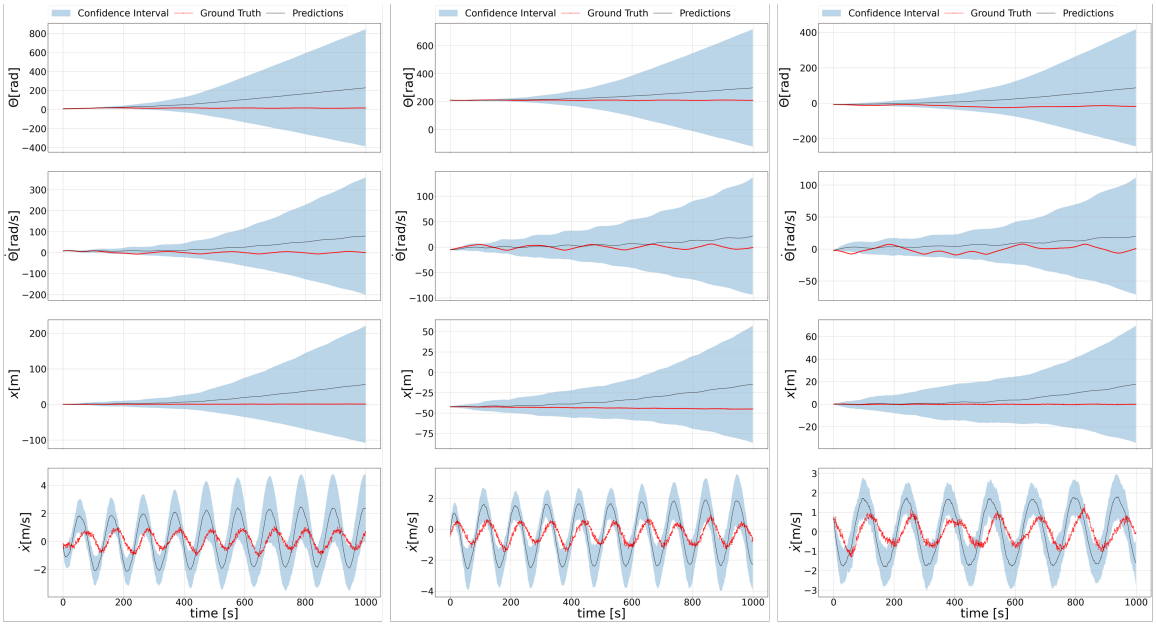

Fig. 5 Multi-step performance of deep ensembles on high noise data (a) Train (b) Validation (c) Test

VIII. Future Directions

In the context of diagnostics, it is imperative to detect a system's behavioral change, especially due to faults during the life cycle of the system. Suppose that, due to a system fault, the system begins producing data that the learned model (deep ensemble) has never seen. In that case, it should be tagged as OOD with increased predictive uncertainty. Unfortunately, it has been noted that the deep ensembles raise their variance when *inaccurate*, not necessarily when the data are OOD [37]. This is a topic of concern in the literature and needs to be evaluated further in the context of diagnostics. The feature can be assessed with cartpole simulation by intentionally creating a distribution shift in the data that mimics faulty system behavior. In future work, we will develop a suite of scenarios to test the hypothesis that deep ensembles are good at UQ when they are inaccurate but not good at detecting OOD situations. We also plan to test several OOD and anomaly detection algorithms on the cartpole problem in the future [40].

IX. Concluding Remarks

In this paper, deep ensembles were used as surrogate transition models for the cartpole ODEs. A neural network, such as multi-layer perceptron could be used for modeling the problem. However, the most significant advantage of deep ensembles is their ability to quantify aleatory and epistemic uncertainty. The deep ensembles were trained based on the data collected by simulating ODEs of the cartpole and demonstrated their effectiveness in estimating aleatory and epistemic uncertainties. Two different testing modes were used: single-step and multi-step. In single-step mode, the model accurately predicted transitions and manifested *mainly* aleatory uncertainty. This claim is made by visually comparing the uncertainty of predictions across different noise levels. On the other hand, the multi-step approach depicted both aleatory and epistemic uncertainties. In multi-step mode, the model's predictions at the previous time step are fed into the model to compute predictions at the next time step. This poses a challenge of uncertainty propagation for future time steps due to the accumulation of errors. The trajectory sampling method **TS**∞ was used for uncertainty propagation. With predictions further into the future, the model's accuracy decreases; hence, the uncertainty increases. Even though the deep ensembles are not ideal for uncertainty quantification, they are by far among the best models for use in decision-making frameworks as surrogate transition models.

References

[1] Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O., "Understanding deep learning requires rethinking generalization," arXiv:1611.03530 [cs.LG], 2017.

[2] Jiang, Y., Neyshabur, B., Mobahi, H., Krishnan, D., and Bengio, S., "Fantastic Generalization Measures and Where to Find Them," arXiv:1912.02178 [cs.LG], 2019.

[3] Shalev-Shwartz, S., and Ben-David, S., *Understanding machine learning: From theory to algorithms*, Cambridge university press, 2014.

[4] Neyshabur, B., Bhojanapalli, S., McAllester, D., and Srebro, N., "Exploring Generalization in Deep Learning," arXiv:1706.08947 [cs.LG], 2017.

[5] Arora, S., Ge, R., Neyshabur, B., and Zhang, Y., "Stronger generalization bounds for deep nets via a compression approach," arXiv:1802.05296 [cs.LG], 2018.

[6] Liu, J., Shen, Z., He, Y., Zhang, X., Xu, R., Yu, H., and Cui, P., "Towards Out-Of-Distribution Generalization: A Survey," arXiv:2108.13624 [cs.LG], 2023.

[7] Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q., "On Calibration of Modern Neural Networks," arXiv:1706.04599 [cs.LG], 2017.

[8] Wald, Y., Feder, A., Greenfeld, D., and Shalit, U., "On Calibration and Out-of-Domain Generalization," *Advances in Neural Information Processing Systems*, edited by A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, 2021. URL `https://openreview.net/forum?id=XWYJ25-yTRS`.

[9] Wilson, A. G., and Izmailov, P., "Bayesian Deep Learning and a Probabilistic Perspective of Generalization," arXiv:2002.08791 [cs.LG], 2022.

[10] Lakshminarayanan, B., Pritzel, A., and Blundell, C., "Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles," arXiv:1612.01474 [stat.ML], 2017.

[11] Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J. V., Lakshminarayanan, B., and Snoek, J., "Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift," arXiv:1906.02530 [stat.ML], 2019.

[12] Gustafsson, F. K., Danelljan, M., and Schön, T. B., "Evaluating Scalable Bayesian Deep Learning Methods for Robust Computer Vision," arXiv:1906.01620 [cs.LG], 2020.

[13] Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., Fieguth, P., Cao, X., Khosravi, A., Acharya, U. R., Makarenkov, V., and Nahavandi, S., "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," *Information Fusion*, Vol. 76, 2021, pp. 243–297. https://doi.org/10.1016/j.inffus.2021.05.008, URL https://doi.org/10.1016%2Fj.inffus.2021.05.008.

[14] Li, Y. L., Rudner, T. G. J., and Wilson, A. G., "A Study of Bayesian Neural Network Surrogates for Bayesian Optimization," arXiv:2305.20028 [cs.LG], 2023.

[15] Rasmussen, C. E., "Gaussian processes in machine learning," *Summer school on machine learning*, Springer, 2003, pp. 63–71.

[16] Wilson, A., and Adams, R., "Gaussian process kernels for pattern discovery and extrapolation," *International conference on machine learning*, PMLR, 2013, pp. 1067–1075.

[17] Gardner, J., Pleiss, G., Weinberger, K. Q., Bindel, D., and Wilson, A. G., "Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration," *Advances in neural information processing systems*, Vol. 31, 2018.

[18] Hensman, J., Fusi, N., and Lawrence, N. D., "Gaussian processes for big data," *arXiv preprint arXiv:1309.6835*, 2013.

[19] Titsias, M., "Variational learning of inducing variables in sparse Gaussian processes," *Artificial intelligence and statistics*, PMLR, 2009, pp. 567–574.

[20] Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J., "Stochastic variational inference," *Journal of Machine Learning Research*, 2013.

[21] Jain, S., Liu, G., Mueller, J., and Gifford, D., "Maximizing Overall Diversity for Improved Uncertainty Estimates in Deep Ensembles," arXiv:1906.07380 [cs.LG], 2020.

[22] Zaidi, S., Zela, A., Elsken, T., Holmes, C., Hutter, F., and Teh, Y. W., "Neural Ensemble Search for Uncertainty Estimation and Dataset Shift," arXiv:2006.08573 [cs.LG], 2022.

[23] He, B., Lakshminarayanan, B., and Teh, Y. W., "Bayesian deep ensembles via the neural tangent kernel," *Advances in neural information processing systems*, Vol. 33, 2020, pp. 1010–1022.

[24] Huang, G., Li, Y., Pleiss, G., Liu, Z., Hopcroft, J. E., and Weinberger, K. Q., "Snapshot Ensembles: Train 1, get M for free," arXiv:1704.00109 [cs.LG], 2017.

[25] Fort, S., Hu, H., and Lakshminarayanan, B., "Deep Ensembles: A Loss Landscape Perspective," arXiv:1912.02757 [stat.ML], 2020.

[26] Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T., "Visualizing the Loss Landscape of Neural Nets," arXiv:1712.09913 [cs.LG], 2018.

[27] Garipov, T., Izmailov, P., Podoprikhin, D., Vetrov, D., and Wilson, A. G., "Loss Surfaces, Mode Connectivity, and Fast Ensembling of DNNs," arXiv:1802.10026 [stat.ML], 2018.

[28] Louizos, C., and Welling, M., "Multiplicative Normalizing Flows for Variational Bayesian Neural Networks," arXiv:1703.01961 [stat.ML], 2017.

[29] Gal, Y., and Ghahramani, Z., "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," arXiv:1506.02142 [stat.ML], 2016.

[30] Goodfellow, I. J., Vinyals, O., and Saxe, A. M., "Qualitatively characterizing neural network optimization problems," *arXiv preprint arXiv:1412.6544*, 2014.

[31] Wilson, A. G., "Deep Ensembles as Approximate Bayesian Inference — cims.nyu.edu," https://cims.nyu.edu/~andrewgw/deepensembles/, 2021. [Accessed 01-12-2023].

[32] Izmailov, P., Vikram, S., Hoffman, M. D., and Wilson, A. G., "What Are Bayesian Neural Network Posteriors Really Like?" arXiv:2104.14421 [cs.LG], 2021.

[33] Cannon, R. H., *Dynamics of physical systems*, Courier Corporation, 2003.

[34] Brunton, S. L., Proctor, J. L., and Kutz, J. N., "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the National Academy of Sciences*, Vol. 113, No. 15, 2016, p. 3932–3937. https://doi.org/10.1073/pnas.1517384113, URL http://dx.doi.org/10.1073/pnas.1517384113.

[35] Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., et al., "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[36] Deisenroth, M., and Rasmussen, C. E., "PILCO: A model-based and data-efficient approach to policy search," *Proceedings of the 28th International Conference on machine learning (ICML-11)*, 2011, pp. 465–472.

[37] Lu, C., Ball, P. J., Parker-Holder, J., Osborne, M. A., and Roberts, S. J., "Revisiting Design Choices in Offline Model-Based Reinforcement Learning," arXiv:2110.04135 [cs.LG], 2022.

[38] Chua, K., Calandra, R., McAllister, R., and Levine, S., "Deep reinforcement learning in a handful of trials using probabilistic dynamics models," *Advances in neural information processing systems*, Vol. 31, 2018.

[39] Depeweg, S., Hernandez-Lobato, J.-M., Doshi-Velez, F., and Udluft, S., "Decomposition of uncertainty in Bayesian deep learning for efficient and risk-sensitive learning," *International Conference on Machine Learning*, PMLR, 2018, pp. 1184–1193.

[40] Yang, J., Wang, P., Zou, D., Zhou, Z., Ding, K., Peng, W., Wang, H., Chen, G., Li, B., Sun, Y., Du, X., Zhou, K., Zhang, W., Hendrycks, D., Li, Y., and Liu, Z., "OpenOOD: Benchmarking Generalized Out-of-Distribution Detection," arXiv:2210.07242 [cs.CV], 2022.

A. Deep Ensembles Results

A. Single-step Performance of Deep Ensembles

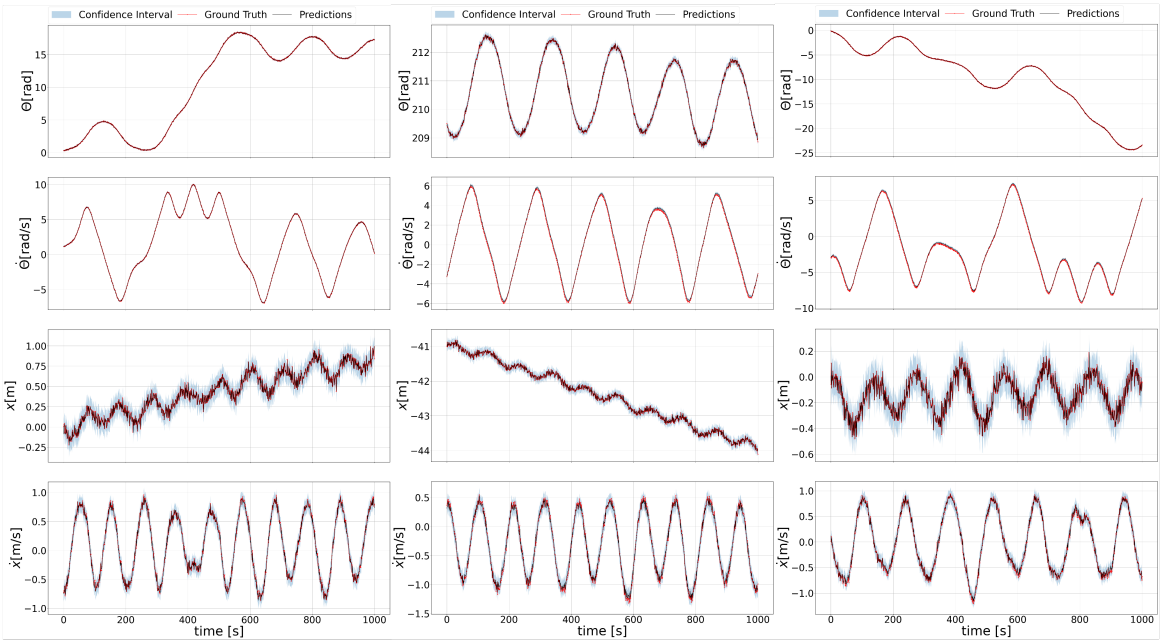

Fig. A.1 Single-step performance of deep ensembles on low noise data (a) Train (b) Validation (c) Test

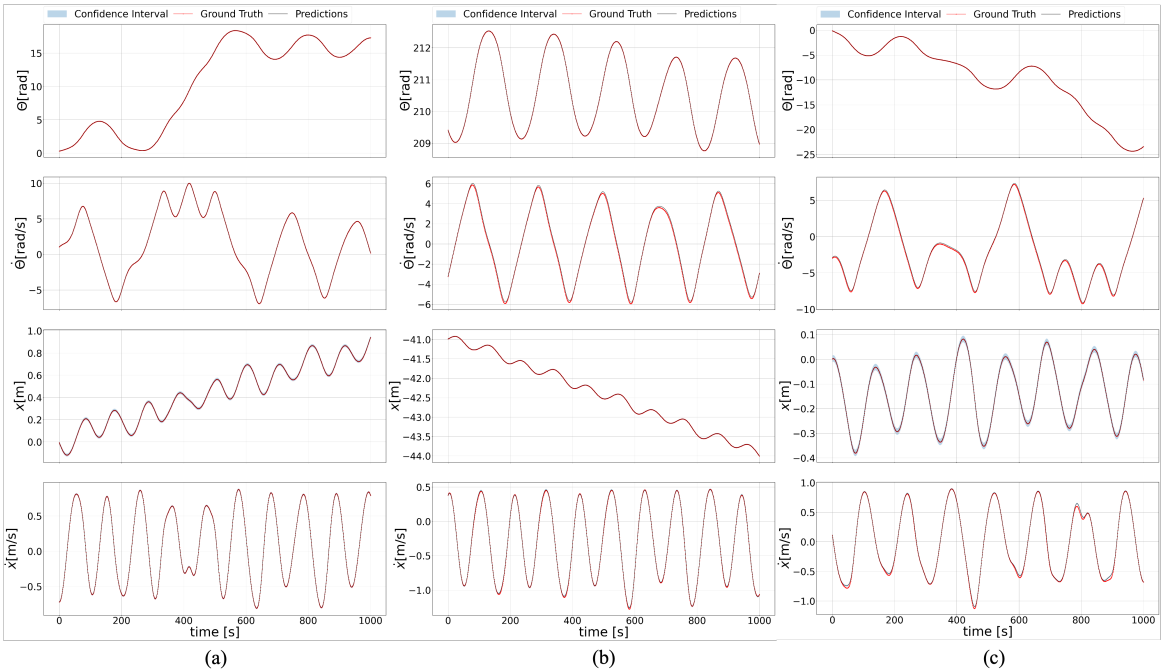

Fig. A.2 Single-step performance of deep ensembles on deterministic data (a) Train (b) Validation (c) Test

B. Multi-step Performance of Deep Ensembles

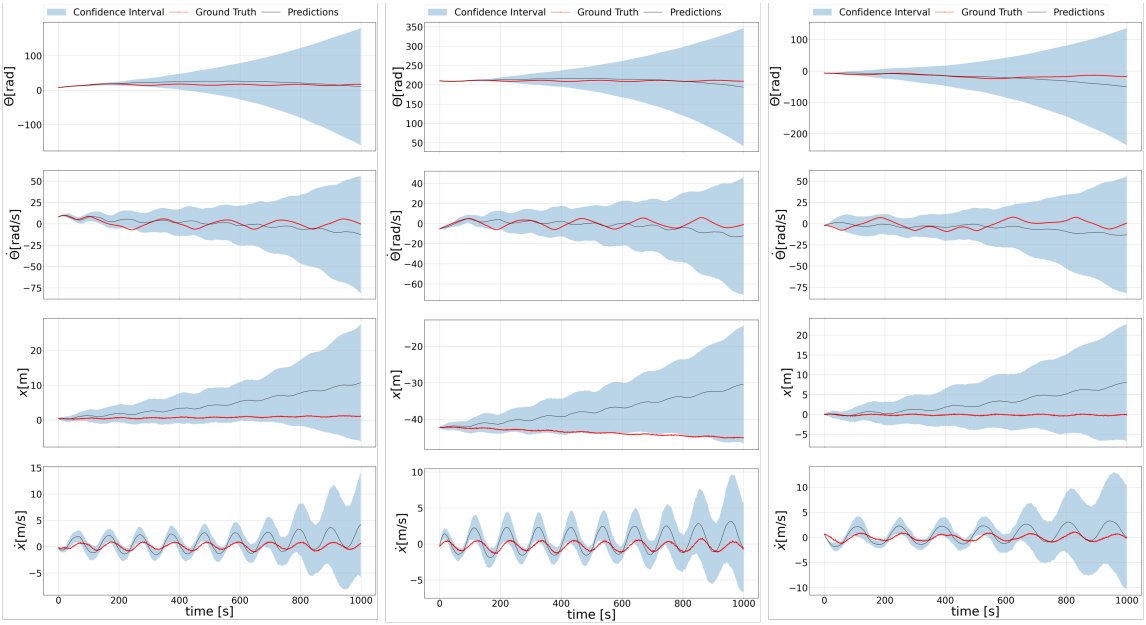

Fig. A.3 Multi-step performance of deep ensembles on low noise data (a) Train (b) Validation (c) Test

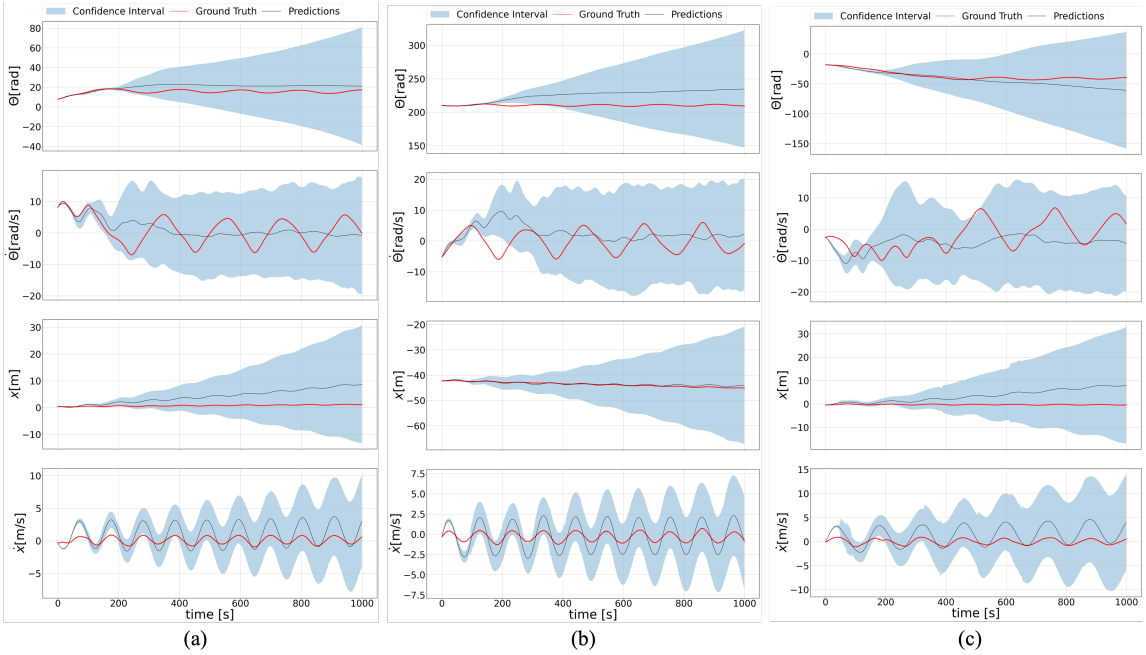

Fig. A.4 Multi-step performance of deep ensembles on deterministic data (a) Train (b) Validation (c) Test