Learning Direct Solution in Moving Horizon Estimation with Deep Learning Methods

Fabien Lionti
Université de Côte d'Azur - INRIA
2004 Rte des Lucioles, 06902 Valbonne
fabien.lionti@inria.fr

Nicolas Gutowski
Université d'Angers - LERIA
2 Bd de Lavoisier, 49000 Angers
nicolas.gutowski@univ-angers.fr

Sébastien Aubin
Direction Générale de l'Armement
Rue de la Chéddittière, 49460 Montreuil-Juigné
sebastien.aubin@intradef.gouv.fr

Philippe Martinet
Université de Côte d'Azur - INRIA
2004 Rte des Lucioles, 06902 Valbonne
philippe.martinet@inria.fr

Abstract— **State estimation in the context of dynamical systems is crucial for various applications, including control and monitoring. Moving Horizon Estimation (MHE) is an optimization-based state estimation algorithm that leverages a known dynamical model integrated over a moving horizon. The MHE optimization criterion corresponds to identify the initial state that best aligns the integrated trajectory with the system observation. In MHE setting, the state estimation performance increases with the considered length of the moving horizon but it can become computationally intensive which is a limiting factor for its applicability to fast-varying dynamical systems or on hardware with restricted computational power. Deep Learning (DL) methods can learn solutions to complex optimization problems without incurring any additional online computational cost beyond the inference of the considered architecture. In the context of state estimation we propose to study different type of DL architecture in order to provide full state estimation from partial and noisy system observations. The novel proposed method is based on an end-to-end differentiable formulation of the MHE optimization problem, enabling the offline training of a DL model to provide a state estimation that minimizes the MHE optimization criterion. Once training is completed, state estimations are generated through an explicit relationship learned by the DL model. The proposed method is compared to the online MHE formulation in various case studies, including scenarios with partially observed state and model discrepancies in the context of lateral vehicle dynamics. The results highlight improved state estimation performance both in terms of reduced computational time and accuracy with respect to the online MHE algorithm.**

Index Terms— **State Estimation, Moving Horizon Estimation, Deep Learning**

I. Introduction

Estimating the state of dynamical systems from measurements has been a persistent challenge in control theory. This complexity stems from various factors such as integration constraints due to sensors, limitations in extreme local conditions, the absence of sensors for specific measurements, or simply cost constraints. Overcoming these obstacles is crucial for the development of effective controllers, especially when dealing with states that cannot be directly measured. The problem of state estimation was initially introduced by Kalman in his seminal paper from 1960 [1]. Kalman extended this study to linear stochastic systems, providing guaranteed optimal state estimation under Gaussian noise in 1961 [2]. Luenberger further contributed to the field by presenting a generalized theory of state observers for linear deterministic systems, introducing the concepts of reduced and minimal state observers [3], expanding the understanding and applicability of state estimation techniques. The prevalence of nonlinear systems has driven the development of state estimation methods beyond linear observers. The Extended Kalman Filter [4] addresses nonlinearity through linearization, but its accuracy is locally limited, leading to potential inaccuracies in the presence of strong nonlinear behavior. In contrast to these approaches, Moving Horizon Estimation (MHE) [5] is an optimization-based state estimation method. In MHE, state estimation is performed by solving a real-time optimization problem that minimizes an approximate optimal control cost function over a defined moving horizon. This method has the advantage of handling various uncertainty distributions while explicitly accounting for physical constraints on both states. Moreover, MHE shows great promise, particularly for nonlinear state estimation, due to its robust stability properties [6]. The performance of MHE improves as the length of the receding horizon increases. However, this also leads to a computational burden proportional to the size of the considered interval horizon. In this work, a methodology is proposed to train a DL model to learn a solution for the MHE optimization problem, specifically for full-state estimation from partial and noisy system observations.

Key contributions of this paper are the following : 1) The proposed method is built on a fully differentiable formulation of the MHE optimization problem, leveraging the automatic differentiation (AD) framework [7]. This enables the DL model to learn a direct relationship that approximates the solution of this problem during an offline training period on a dataset containing system observations. 2) After the training phase, the DL model can provide state estimation with significantly improved computation time compared to the online MHE (OMHE) algorithm by not relying on any additional optimization process. 3) The training process leverages modern DL inductive bias to estimate solutions that globally minimize the MHE criterion across the entire dataset at all time steps, eliminates the need for arrival cost terms as in the OMHE algorithm to maintain consistency in the sequence estimation over time. Experimental results show improved performance on complex cases, including those with model discrepancies, compared to the OMHE algorithm.

II. Related work

Recently breakthroughs in DL algorithms have sparked a growing interest in applying these methods within the domain of dynamical systems. In the realm of state estimation, KalmanNet [8] is a neural network-based state estimator that learns from data within a Kalman filtering framework. It addresses nonlinear dynamics and partial information by incorporating the structure of a state space model into a recurrent neural network (RNN) architecture. KalmanNet effectively manages nonlinearities and model mismatches, outperforming traditional filtering methods in situations with both accurate and inaccurate domain knowledge. Similarly, Backprop KF [9] leverages RNNs within a probabilistic state filtering framework, enabling the design of network architectures specifically tailored for state estimation. The method is designed to train state estimators that utilize complex input sensors, such as images, and demonstrates significant improvements over both standard generative approaches and regular RNNs. In MHE context, [10] addresses the problem of online state estimation and parameter tuning for constrained linear systems by employing a differentiable convex optimization layer to formulate a MHE state estimator using stochastic gradient descent (SGD). Their method is suited for the case of constrained linear systems with parametric uncertainty and show improvement in the performance with respect to the OMHE formulation. In a closely related context outlined in [11], a two-step data-driven MHE-based framework is introduced. The authors leverage an AutoEncoder to construct a surrogate model capable of preserving the states of complex and computationally demanding digital twins. Subsequently, the AutoEncoder is harnessed for online state estimation from noisy measurements. Recently, a paradigm shift in solving inverse problems has emerged through the application of machine learning (ML) models. These models can learn to optimize or directly provide solutions to optimization problems, thereby improving the accuracy and computational efficiency of estimates. In this context, the MHE method involves estimating the state by solving an inverse problem based on an implicit relationship between the initial state of an initial value problem (IVP) and a sequence of system observations. Identifying the initial state requires minimizing this implicit formulation using an iterative solver, which is known to be time-consuming. Following this paradigm, there has been growing interest in using ML models to learn approximate solutions to the MHE problem. In [12] a one-layer perceptron is estimated by solving a nonlinear programming problem in order to learn to minimize in offline manner an MHE optimization problem. In [13] multi-layer perceptron is used to learn approximate solutions of simultaneously solution for the MHE and Model Predictive Control (MPC) problems using data generated by OMHE and MPC algorithms in order to significantly reduce computational cost to obtain state estimates. In a related direction, the proposed method in this paper rely on a fully differentiable formulation of the MHE optimization criterion allowing to learn to a DL model to infer MHE solution directly from measurement data without relying on OMHE estimation such as in [13]. In order to evaluate the influence of modern DL inductive biases on estimated performance, a comprehensive comparison between different DL models is conducted.

III. Proposed method

A general system described by a finite dimensional continuous non-linear dynamic of the form is considered

$$\dot{x}(t) = f(x(t), u(t)) \tag{1}$$

$$y(t) = g(x(t), u(t)) + \eta \tag{2}$$

with state $x \in \mathbb{R}^n$, control $u \in \mathbb{R}^m$, observation $y \in \mathbb{R}^p$, and smooth maps : $f : \mathbb{R}^n \to \mathbb{R}^n$ corresponding to the dynamical system equations, measurement function $g : \mathbb{R}^n \to \mathbb{R}^p$ and η a random variable corresponding to noise measurement. An initial strategy for state estimation involves simulating the equations f concurrently with a range of likely initial states chosen a priori, and iteratively discarding those that result in an output trajectory $y(t)$ that deviates significantly from the observed one. If the output $y(t)$ uniquely determines a solution asymptotically, the set of feasible initial states converges to a single possibility. Conversely, if multiple initial states persistently yield feasible solutions, this situation corresponds to a system with non-observable states. In this paper, the focus is on system where the state is globally observable. Based on this principle, the state estimation problem can be formulated as an optimization problem with the following criterion:

$$\mathcal{L}(x(t_0)) = \int_{t_0}^{t_0+T_f} \Big(\hat{y}(x(t_0), f, u, \tau) - y(\tau)\Big)^2 d\tau \tag{3}$$

$$x^*(t_0) = \min_{x(t_0)} \mathcal{L}(x(t_0)) \tag{4}$$

with x^* that corresponds to the initial state that minimizes the difference between the simulated $\hat{y}$ and the measured state variable trajectory y. This minimization problem is solved over a time horizon $\tau \in [t_0, T_f]$ obtained from the solution of an IVP involving the system dynamic f under the influence of the control input u:

$$\hat{y}(x(t_0), f, u, \tau) = g\Big(x(t_0) + \int_{t_0}^{t_0+\tau} f(x(t), u(t))dt\Big) \tag{5}$$

The proposed method aims to recover all state variables from noisy and partial system measurements y, implying that $n > p$, by using a DL model ϕ that learns to predict the initial state estimation $x(t_0)$, minimizing $\mathcal{L}$ over the future horizon T_f :

$$x(t_0) = \phi\Big(y(t_0), y(t_0 - \tau_p), \ldots, y(t_0 - L\tau_p; \theta\Big) \tag{6}$$

with L the total number of past samples considered, such that $T_p = t_0 - L\tau_p$ corresponds to the past horizon, and

θ representing the parametrization of the DL model. For training purposes, the loss function (3) is reformulated in a discretized version:

$$\mathcal{L}(\theta) = \frac{1}{MK} \sum_{m=0}^{M} \sum_{k=0}^{K} \Big(\hat{y}(\phi, f, u, t_0 + k\tau_f) - y(t_0 + k\tau_f) \Big)^2 \quad (7)$$

where τ_f is a future delay separating each sample and $T_f = t_0 + K\tau_f$ and M corresponds on the number training trajectory available in the dataset $\mathcal{D}$. State trajectory estimates $\hat{y}$ are obtained using numerical integration method :

$$\hat{y}(\phi, f, u, t_0 + k\tau_f) = g \circ F^k \circ F^{k-1} \circ \cdots \circ F^0(x(t_0), u(t_0)) \quad (8)$$

with $F(x(t), u(t)) = x(t + \tau_f)$ corresponding to a one-step forward Euler integration method starting from $x(t_0)$:

$$x(t + \tau_f) = f(x(t), u(t))\tau_f + x(t) \quad (9)$$

By implementing f and g model under PyTorch framework, it allows to build a computational graph of the numerical integration schema, and, by using AD algorithm to makes the MHE optimization problem fully differentiable in order to obtain gradient with respect to parametrization θ by backpropagating gradient over time from $t_0 + k\tau_f$ to t_0 :

$$\frac{\partial \hat{y}(\phi, f, u, t_0 + k\tau_f)}{\partial \theta} = \frac{\partial g}{\partial F^k} \prod_{i=0}^{i=k-1} \frac{\partial F^{i+1}}{\partial F^i} \frac{\partial F^0}{\partial \phi} \frac{\partial \phi}{\partial \theta} \quad (10)$$

The proposed method differs from the OMHE algorithm as there is no arrival cost in the loss (7). In the sequential context of the OMHE algorithm, this cost is used to penalize the difference between the previous estimate and the current one. This allows for maintaining consistency between successive estimates and "virtually" increasing the length of the moving window by re-injecting prior state estimate knowledge from the past to the current estimate. The proposed method transforms the temporally localized optimization problem at a current time step in the OMHE context into an optimization problem that operates at all time steps available in $\mathcal{D}$. In this case, adding an arrival cost is unnecessary because the DL model inherently learns to preserve continuity across its predictions by minimizing loss (7) over the entire dataset $\mathcal{D}$. Additionally, the proposed fully-differentiable MHE loss eliminates the need to supervise the training directly with state estimation generated by the OMHE algorithm, as in [13], and extends the method to more efficient optimization techniques than in [12], such as gradient descent, allowing for the use of more complex DL models. Another motivation for the MHE formulation, compared to other state estimators, is the ability to apply constraints on state estimation by adding penalization terms to the optimization criterion. Although not covered in this paper, the imposition of constraints on the estimated state can be easily extended by adding terms to the loss function (7).

IV. Experimental setting

This section describes the experimental settings of the realized numerical study in order to evaluate the performance of the proposed method. This study is performed over various configurations of future T_f : $\{0.1, 0.2, 0.4, 0.8\}$ and past T_p : $\{1, 2, 3, 4\}$ horizons. It includes different DL models, specifically one-dimensional convolutional neural networks (CNN) [14] choosen for their capability to effectively captures local patterns and temporal dependencies at different scales, gated recurrent unit (GRU) [15] for their capability to capture sequential dependencies and maintain information across time steps, and a reduced and 1D version of CNN transformers (CNN-T) models allowing for handling of complex temporal relationships such as in [16]. All experiments are performed with corresponding parameters $\tau_p = 0.02$ and $\tau_f = 0.02$.

Case study description : Case studies corresponding to the state estimation task from noisy and partial system measurement data generated by the Van der Pol [17] and by the forced Duffing oscillators [18] are included. Both of these systems have a two-dimensional state vector $x = [x_0, x_1]^T$ with partial and noisy observability conditions corresponding to $g = \text{diag}([1, 0])x + \eta$. In the case of the Duffing oscillator, the forcing function corresponds to the control input $u(t) = 0.3\cos(t)$ and η is zero-mean Gaussian noise with a standard deviation $\sigma = 0.45$. For the Van der Pol oscillator, the Gaussian noise has a standard deviation $\sigma = 1.5$. In both cases, the model f used for the training phase is similar to the one used to generate the training and test datasets. Initial conditions for each state variable of the simulated trajectory are uniformly chosen from the interval $[-10, 10]$ in the Van der Pol case, and from $[-1, 1]$ in the Duffing case.

A third case study on complete but noisy measurement data in the context of lateral vehicle dynamics is included. It corresponds to the state estimation of the lateral speed x_0 in $m \cdot s^{-1}$ and the yaw rate x_1 in $rad \cdot s^{-1}$ of the vehicle, with an observation function $g = \text{diag}([1, 1])x + \eta$. In this case study, data are generated from a ten-degree-of-freedom vehicle model (Dof10) that includes longitudinal, lateral, and suspension dynamics. The model f used for the training phase corresponds to a two-degree-of-freedom (Dof2) lateral bicycle model that neglects both longitudinal and mass transfer dynamics [19]. This represents a case with complex nonlinear coupling between simulated state variables and includes model discrepancy during training. The control input u applied to the Dof2 f model in this case is the measured longitudinal speed v_x in $m \cdot s^{-1}$ and the steering angle α in rad applied to the Dof10 model for data simulation. Trajectories are simulated using random initial longitudinal speeds v_x uniformly sampled from the interval $[5, 25]$, constant front wheel torque in $N \cdot m$ uniformly sampled from the interval $[50, 100]$, and random sinusoidal steering α with amplitude uniformly sampled from $[-0.0872, 0.082]$, corresponding to $\pm 10°$ steering angle in degrees. A two-dimensional uncorrelated Gaussian noise η is added to each state variable, with a standard deviation $\sigma = 0.1\, m \cdot s^{-1}$ for the lateral speed x_0 and $\sigma = 1\, deg \cdot s^{-1}$ for the yaw rate x_1.

Dataset generation : For each case study, the datasets are generated from the numerical integration of equations describing the dynamical behavior of each previously mentioned system with an integration step of 10^{-3} seconds, 200 trajectories are simulated for the Van der Pol and the Forced Duffing, 300 trajectories are generated from the Dof10 vehicle

	Method	**CNN**		**GRU**		**CNN-T**		**OMHE**	
Case study	**Horizon (T_p)**	x_0	x_1	x_0	x_1	x_0	x_1	x_0	x_1
Van Der Pol	**0.5**	0.3687 ± 0.003	0.6715 ± 0.0093	**0.3681 ± 0.0038**	**0.6651 ± 0.01**	0.395 ± 0.0063	0.754 ± 0.0566	0.533	1.2407
	1.0	0.3034 ± 0.005	**0.3386 ± 0.0023**	**0.2936 ± 0.005**	0.3508 ± 0.0095	0.3713 ± 0.0128	0.4494 ± 0.0247	0.3517	0.5632
	2.0	0.2322 ± 0.0021	**0.1786 ± 0.0119**	**0.2288 ± 0.0026**	0.1853 ± 0.0077	0.3236 ± 0.0256	0.3031 ± 0.029	0.2385	0.261
	4.0	0.1709 ± 0.0061	0.1127 ± 0.0016	**0.1503 ± 0.0022**	0.1064 ± 0.0071	0.2891 ± 0.0227	0.2082 ± 0.0354	0.1657	**0.0689**
Duffing	**0.5**	0.1405 ± 0.003	0.4129 ± 0.0104	**0.1345 ± 0.0002**	**0.385 ± 0.0011**	0.1836 ± 0.018	0.4549 ± 0.0202	0.1759	0.5655
	1.0	0.1075 ± 0.0026	0.1677 ± 0.0026	**0.1054 ± 0.001**	**0.1626 ± 0.0024**	0.1565 ± 0.0167	0.268 ± 0.0567	0.1164	0.1843
	2.0	**0.0665 ± 0.0019**	0.1078 ± 0.0029	0.0667 ± 0.0005	**0.0988 ± 0.0039**	0.1035 ± 0.0268	0.2473 ± 0.1426	0.0706	0.106
	4.0	0.0553 ± 0.0006	0.0893 ± 0.0044	**0.0536 ± 0.0003**	0.0869 ± 0.0037	0.0957 ± 0.0352	0.1424 ± 0.0791	0.0546	**0.0772**
Dof10	**0.5**	0.0998 ± 0.024	0.0386 ± 0.0121	**0.0589 ± 0.0053**	**0.015 ± 0.0031**	0.1757 ± 0.1035	0.0545 ± 0.0506	0.2434	0.0226
	1.0	0.0987 ± 0.015	0.0257 ± 0.0112	**0.068 ± 0.0045**	**0.0148 ± 0.0021**	0.1828 ± 0.1362	0.0543 ± 0.0292	0.3907	0.0324
	2.0	0.0796 ± 0.0077	0.0188 ± 0.002	**0.0656 ± 0.0048**	**0.0136 ± 0.0019**	0.1465 ± 0.044	0.0425 ± 0.0056	0.4832	0.0376
	4.0	0.1015 ± 0.0085	0.0215 ± 0.0027	**0.0786 ± 0.0104**	**0.0106 ± 0.0008**	0.1413 ± 0.0542	0.0463 ± 0.0114	0.4963	0.0384

TABLE I: This table describes the RMSE performance for each case study, deep learning architecture, state variable, and past temporal horizon window. The RMSE values in this table are associated with the best future temporal horizons T_f: 0.8 for the Van der Pol system, 0.8 for the Duffing system, and 0.1 for the Dof10 case study. RMSE values are provided with uncertainty corresponding to ± one standard deviation.

model with a duration $T = 20$ seconds. Training, validation, and test datasets are defined by randomly splitting trajectories according to the following proportions: 70%, 20%, and 10%.

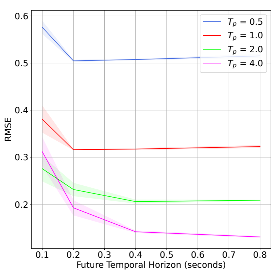
Evaluation method : The Root Mean Squared Error (RMSE) is used as the performance metric to measure the prediction error between the noise-free estimates $\hat{y}$ and the estimated values y. To assess the robustness of the proposed method, each previously mentioned experimental configuration is repeated three times with different DL model parameter initializations. As the method is based on MHE optimization principle, we restricted the scope of comparison to the OMHE algorithm using the efficient HILO-MPC Python implementation based on Casadi and Ipopt solver [20]. The MHE weighting parameters associated with the quadratic arrival cost, measurements noise cost and state noise cost, have been hand-tuned to ensure the best possible performance. The implementation and experimentation details of the proposed method are available on GitHub at https://github.com/N9TT-9G0A-B7FQ-RANC/ICRA_2025

V. Results

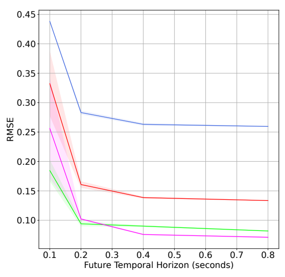
Table I presents the RMSE computed for each state variable across all experimented T_p values, along with the associated T_f value that gave the lowest RMSE on the test set. Among the tested architectures, the GRU is globally associated with the lowest RMSE across all case studies, closely followed by the CNN. This can be interpreted as the adequacy of the GRU inductive bias, which is particularly well-suited for the sequential nature of the problem. Although the CNN provides less consistent performance, it also seems to be a good candidate as it sometimes outperforms the GRU architecture. The CNN-T architecture is associated with significant performance degradation, which can be interpreted by the fact that the attention inductive bias tends to partially lose temporal information in its internal representation, providing less accurate results.

Two patterns can be identified in the evolution of the RMSE for different values of past horizons. The first pattern corresponds to an improvement in performance for the proposed method and the OMHE algorithm, induced by increasing the past horizon in the Van der Pol and Duffing case studies. The second pattern is associated with a stagnation in performance for the proposed method and a significant decrease in performance for the OMHE algorithm when increasing T_p in the Dof10 case study. In the first pattern, the Van der Pol and Duffing oscillators correspond to cases where the model used during training and in the OMHE algorithm is identical to the one used to simulate the training and test data. In this case, the CNN and GRU architectures learn to estimate states with slightly better performance than the OMHE method. In the second pattern, the Dof10 case study illustrates a situation where there is a discrepancy between the model used in the proposed method and the OMHE algorithm, compared to the one used to simulate the training and test datasets. In this scenario, the proposed method significantly outperforms the OMHE algorithm. Results indicate that for the OMHE algorithm, increasing the temporal horizon is associated with an increase in error, whereas for the proposed method, the error remains stagnant. This phenomenon underscores the fundamental difference between the proposed method and the OMHE method.

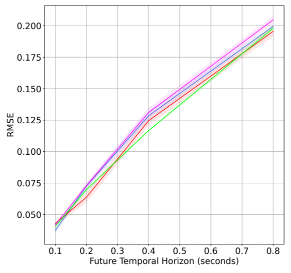
The Figure 1 illustrates the evolution of the averaged RMSE for each state variable across different experimented T_f and T_p values for the GRU model. The two RMSE patterns previously mentioned can be identified: one for the Van der Pol 1a and Duffing 1b case studies, and the second for the Dof10 1c case study, which corresponds to a model mismatch. In the first evolution pattern, for the proposed method, employing the longest T_p and T_f is associated with a reduction in RMSE. A convergence phenomenon can be observed. After a certain T_f value, increasing the integration time does not seem to provide significant performance improvement. This phenomenon can be interpreted as follows: as there is no model mismatch, the model must learns a representation that contains enough past temporal information to estimate the unobserved state variable. To learn a robust representation from this past window and fully exploit the information contained in this sequence, longer integration times during training are beneficial. This allows for optimizing the MHE loss over longer state variable trajectories, resulting in a gradient from the loss function that is more robust to noise fluctuations. This can also be perceived by the associated variance reduction with a higher T_f parameter. For intermediate T_p values, which correspond to a situation where the quantity of information fed into the architecture is reduced, the intrinsic capability of the



(a) Van Der Pol



(b) Duffing



(c) Dof10

Fig. 1: This figure provides RMSE for each T_p and T_f values on each case study for the GRU architecture

model to learn a robust representation is more constrained. Although increasing the integration time improves the quality of the gradient during the training phase, the model is not complex enough to benefit from this increase, leading to an earlier stagnation point in performance compared to cases with longer T_f values. The second pattern is associated with the Dof10 case study, where there is a discrepancy between the model employed during training and the one used to simulate state variable trajectories. Increasing the T_f parameter is associated with an increase in RMSE. During the training phase, the DL model learns initial state estimates by minimizing the difference between trajectories generated from a biased model and those used to generate measurements. Due to this bias, it is not possible to benefit from longer T_f values, and consequently, it is not possible for the DL model to learn to exploit information for longer T_p horizons.

The Figure 2 illustrates the previous discussions related to Table I and Figure 1. It showcases state estimation over time in the Dof10 case study with the GRU model. The proposed method demonstrates significant noise attenuation, providing smoother and less biased state estimation capabilities than the OMHE algorithm. Although both methods are based on similar optimization criteria, the OMHE method must ensure consistency between the previous and current state estimates using additional arrival cost terms in the optimization process. In the case of using a biased model such as in the Dof10 case study, this continuity constraint can amplify the bias in estimates. Moreover, the exploited model could be locally biased depending on the current state localization, meaning that the weighting of this cost could vary over time with respect to the current model bias to achieve improved performance. The advantage of the proposed method is that

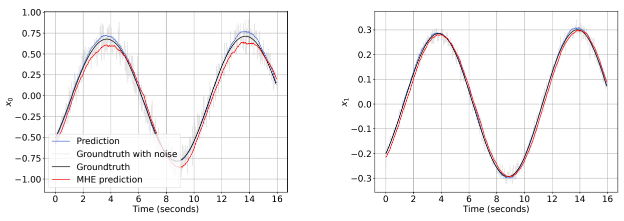


Fig. 2: This figure shows the state estimation capabilities over time for the proposed method, for the lateral speed x_0 and yaw rate x_1 variable in the Dof10 case study.

it does not rely on this arrival cost, allowing the model to learn to maintain maximum smoothness in its predictions and minimal bias by being trained on an entire dataset $\mathcal{D}$ that is representative of the underlying state estimation problem.

The table II shows the computation time, in milliseconds, required to obtain a state estimation for different model architectures compared to the OMHE algorithm, the experiments were conducted on a machine equipped with an 12th Gen Intel Core i9-12900H. In accordance to [12] and [13], results show that leveraging DL model to learn an approximate solution of the MHE problem can significantly reduce the online computational cost associated with obtaining state estimates. The proposed method offers nearly constant prediction times across all case studies, in contrast to the OMHE algorithm which depends both on the complexity of the numerical integration time of the system f and on the online solver used to optimize MHE criterion. Depending on the case study, and the DL model, the proposed approach can reduce computation time by a factor ranging from 10 to 100 compared to the OMHE algorithm. This, combined with the previously discussed performance in terms of RMSE, highlights the main advantage of the proposed method, which allows for the use of DL architectures to achieve accurate and near-instantaneous state estimation from complex mechanical system such as the lateral vehicle dynamic and from partial observation such as in the case of the Van der Pol and Duffing systems.

Limitation : Despite the promising results discussed, the proposed approach has several limitations that should be acknowledged. First, the method must relies on a sufficient quantity of data to ensure generalization capabilities across the

		Method			
Case study	**Horizon (T_p)**	**CNN**	**GRU**	**CNN-T**	**OMHE**
Van Der Pol	**0.5**	**0.0004**	0.0023	0.001	0.1479
	1.0	**0.0004**	0.005	0.001	0.204
	2.0	**0.0003**	0.0078	0.001	0.3197
	4.0	**0.0003**	0.016	0.0014	0.559
Duffing	**0.5**	**0.0004**	0.0026	0.0013	0.1209
	1.0	**0.0003**	0.0041	0.0015	0.1555
	2.0	**0.0003**	0.008	0.0014	0.2295
	4.0	**0.0003**	0.015	0.0017	0.4157
Dof10	**0.5**	**0.0004**	0.0024	0.0012	0.1155
	1.0	**0.0003**	0.0067	0.001	0.1599
	2.0	**0.0003**	0.0087	0.0011	0.249
	4.0	**0.0005**	0.0146	0.0011	0.4328

TABLE II: This table provides a comparison of computation time, in milliseconds, for the different deep learning models and online MHE across each case study and past horizons.

entire state and control space. This requirement assumes that the underlying system is time-invariant, which is necessary to prevent the learned relationships from becoming invalid over time. Additionally, the scope of the proposed method is restricted to deterministic state estimation, providing only point estimates over time. While this may be sufficient for some applications, it does not account for the uncertainty in state estimates, which is crucial in certain use cases. Future research could explore the incorporation of probabilistic DL model within the proposed training scheme to provide uncertainty estimates. Another limitation is related to the potential bias introduced when using an a priori known model that is not fully accurate with respect to the observations. This bias can lead to inaccuracies in the state estimates. To address this, a promising research direction could involve developing a fully data-driven training pipeline where both the initial state and the integrated model are learned directly from data, thereby reducing dependence on potentially biased models.

VI. Conclusion

This paper introduces a novel approach to training deep learning (DL) models to predict state estimation that approximates the solution of a Moving Horizon Estimation (MHE) optimization problem for nonlinear dynamical systems. The method leverages inductive biases from recent DL models to learn a model that takes as input a past and partial horizon window of system measurements and outputs full-state estimation. The learning process is based on a fully differentiable loss function that corresponds to the MHE optimization problem, allowing the DL model to be trained with improved performance in terms of state estimation accuracy and computation time across different case studies compared to the classical MHE algorithm. Even without relying on arrival cost terms, the training formulation optimizes the MHE criterion over all available data, ensuring continuity in predictions and minimizing bias in the estimates. Moreover, the learned relationship allows for quasi-instantaneous state estimation without relying on the a priori known dynamical model, reducing computation time by up to a hundredfold compared to the online MHE algorithm. Although we did not assess the proposed method using data from real-world systems, the results obtained from leveraging a two-degree-of-freedom vehicle model on simulation data derived from the ten-degree-of-freedom vehicle model, provide a scenario with discrepancies similar to those encountered in real-world applications. Moreover, Gaussian noise was applied to all measurements, with variance exceeding that typically in embedded sensors. These factors, combined with the flexibility of the training formulation provide a strong foundation for future research to extend the proposed method to real-world case study and to other estimation tasks in dynamical system context such as parameter estimation.

Acknowledgment

This research is funded by the French Ministry of Defense through the Defense Innovation Agency (AID) and by the National Institute for Research in Digital Science and Technology (INRIA).

References

[1] R. E. Kalman, “A new approach to linear filtering and prediction problems,” *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, Journal of Basic Engineering, pp. 35–45, 1960.

[2] R. E. Kalman and R. S. Buch, “New results in linear filtering and prediction theory,” *Transactions of the ASME Journal of Basic Engineering*, vol. 83, pp. 95–108, 1961.

[3] D. G. Luenberger, “Observing the state of a linear system,” *IEEE Transactions on Military Electronics*, vol. 8, no. 2, pp. 74–80, 1964.

[4] R. E. Kalman, “Signal processing, sensor fusion, and target recognition vi,” in *Proceedings Volume 3068*. Orlando, FL, United States: International Society for Optics and Photonics, 1997.

[5] H. Michalska and D. Mayne, “Moving horizon observers and observer-based control,” *IEEE Transactions on Automatic Control*, vol. 40, no. 6, pp. 995–1006, 1995.

[6] J. D. Schiller, S. Muntwiler, J. Köhler, M. N. Zeilinger, and M. A. Müller, “A lyapunov function for robust stability of moving horizon estimation,” *IEEE Transactions on Automatic Control*, vol. 68, no. 12, pp. 7466–7481, 2023.

[7] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, “Automatic differentiation in machine learning: a survey,” *J. Mach. Learn. Res.*, vol. 18, no. 1, p. 5595–5637, jan 2017.

[8] G. Revach, N. Shlezinger, X. Ni, A. L. Escoriza, R. J. G. van Sloun, and Y. C. Eldar, “Kalmannet: Neural network aided kalman filtering for partially known dynamics,” *Trans. Sig. Proc.*, vol. 70, p. 1532–1547, jan 2022.

[9] T. Haarnoja, A. Ajay, S. Levine, and P. Abbeel, “Backprop kf: Learning discriminative deterministic state estimators,” in *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29. Curran Associates, Inc., 2016.

[10] S. Muntwiler, K. P. Wabersich, and M. N. Zeilinger, “Learning-based moving horizon estimation through differentiable convex optimization layers,” in *Proceedings of The 4th Annual Learning for Dynamics and Control Conference*, ser. Proceedings of Machine Learning Research, R. Firoozi, N. Mehr, E. Yel, R. Antonova, J. Bohg, M. Schwager, and M. Kochenderfer, Eds., vol. 168. PMLR, 23–24 Jun 2022, pp. 153–165.

[11] A. Chakrabarty, A. P. Vinod, H. Mansour, S. A. Bortoff, and C. R. Laughman, “Moving horizon estimation for digital twins using deep autoencoders,” *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 5500–5505, 2023, 22nd IFAC World Congress.

[12] A. Alessandri, M. Baglietto, G. Battistelli, and R. Zoppoli, “Moving-horizon state estimation for nonlinear systems using neural networks,” in *47th IEEE Conference on Decision and Control*, 2008, pp. 2557–2562.

[13] B. Karg and S. Lucia, “Approximate moving horizon estimation and robust nonlinear model predictive control via deep learning,” *Computers Chemical Engineering*, vol. 148, p. 107266, 2021.

[14] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[15] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” in *NIPS Workshop on Deep Learning*, 2014.

[16] H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, and L. Zhang, “Cvt: Introducing convolutions to vision transformers,” in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 22–31.

[17] M. Tsatsos, “Theoretical and numerical study of the van der pol equation,” Ph.D. dissertation, Aristotle University of Thessaloniki, Thessaloniki, 2006.

[18] A. H. Nayfeh and N. E. Sanchez, “Bifurcations in a forced softening duffing oscillator,” *International Journal of Non-Linear Mechanics*, vol. 24, no. 6, pp. 483–497, 1989.

[19] H. Pacejka, *Tyre and Vehicle Dynamics*, ser. Automotive engineering. Butterworth-Heinemann, 2006.

[20] J. Pohlodek, B. Morabito, C. Schlauch, P. Zometa, and R. Findeisen, “Flexible development and evaluation of machine-learning-supported optimal control and estimation methods via HILO-MPC,” 2022.