Adaptive Meta-Learning-Based KKL Observer Design for Nonlinear Dynamical Systems

Lukas Trommer[1], Halil Yiğit Öksüz[2]

Abstract— The theory of Kazantzis-Kravaris/Luenberger (KKL) observer design introduces a methodology that uses a nonlinear transformation map and its left inverse to estimate the state of a nonlinear system through the introduction of a linear observer state space. Data-driven approaches using artificial neural networks have demonstrated the ability to accurately approximate these transformation maps. This paper presents a novel approach to observer design for nonlinear dynamical systems through meta-learning, a concept in machine learning that aims to optimize learning models for fast adaptation to a distribution of tasks through an improved focus on the intrinsic properties of the underlying learning problem. We introduce a framework that leverages information from measurements of the system output to design a learning-based KKL observer capable of online adaptation to a variety of system conditions and attributes. To validate the effectiveness of our approach, we present comprehensive experimental results for the estimation of nonlinear system states with varying initial conditions and internal parameters, demonstrating high accuracy, generalization capability, and robustness against noise.

I. INTRODUCTION

In many control applications, it might not always be feasible to directly measure all states of a dynamical system. In such scenarios, a reliable estimation of these unmeasurable state variables becomes paramount and is usually accomplished through state observers. While the design principles of *Luenberger* observers [1] are commonly applied to linear systems, addressing this challenge for nonlinear systems is inherently more complex [2]. Numerous efforts have been directed to develop nonlinear observers using extended Kalman filters [3]–[5]. However, most of these methods depend on linearization techniques, and therefore, they offer only local convergence.

The theory of *Kazantzis-Kravaris/Luenberger* (KKL) observers [6] extends the Luenberger observer design principles to nonlinear systems. KKL observers are founded on the existence of an injective mapping and its left inverse, enabling the transformation of a nonlinear system into a set of first-order linear differential equations subject to the injection of the nonlinear system output. The existing literature provides comprehensive analyses of KKL observers, including the exploration of requirements for the existence of an injective transformation, discussions on the dimensionality of

the linear state space [7], [8], and the extension of these principles to non-autonomous systems [9], [10]. Although it has been demonstrated that the use of an approximation for the state space transformation is sufficient, deriving a concrete formulation of the corresponding mapping and its left inverse for generic problem statements remains a formidable challenge [11]. One viable approach is to leverage data-driven methods, involving the collection of data samples through system simulation or measurement. With increased computational power and a growing interest in machine learning, neural networks have emerged as versatile solutions for addressing highly complex and nonlinear problems in a variety of domains [12], [13], since they are recognized as universal approximators when appropriately dimensioned [14]. Various strategies have been proposed to formulate the approximation of KKL observer transformation maps as regression problems solvable through feedforward neural networks. In [10], the nonlinear system and the linear part of the observer have been simulated forward in time, starting from pre-specified initial system conditions. Pairwise sets of state data, where one state functions as the input and the other as the label for the nonlinear regression problem, have been generated in order to determine the respective transformation maps. In [15], a deep auto-encoder has been introduced for systems in discrete time, which learns the transformation maps by enforcing observer dynamics in a latent linear space and introducing a measure of reconstruction loss following the composite application of forward and inverse transformations during training. An unsupervised learning technique has been utilized in [16], whose aim is to improve convergence accuracy by using a neural network-based ensemble learning approach. By introducing a physics-informed loss component during training, some improvements in robustness, accuracy, and generalization capabilities have been observed in [17].

It is important to emphasize that learning-based approaches for KKL observer design require labeled, nonlinear system state data for training. While the estimation capabilities of these data-driven approaches benefit from substantial datasets, it is not always practical to utilize arbitrary quantities of data for training. Moreover, in certain scenarios requiring accurate state estimation over varying system parameters, training data may only be available for a limited subset of parameter values. The imposed learning challenge in that case furthermore includes the aim for strong model generalization, allowing accurate estimation even in parameter ranges for which no data is accessible during training.

Aiming to enhance these capabilities, we resort to *Model-*

[1] L. Trommer is with the Control Systems Group at the Technical University of Berlin, Germany. {`lukas.trommer@campus.tu-berlin.de`}

[2] H. Y. Öksüz is with the Control Systems Group and Science of Intelligence, Research Cluster of Excellence, at the Technical University of Berlin, Germany. {`oksuz@tu-berlin.de`}

Agnostic Meta-Learning (MAML), which facilitates training a model on a multitude of tasks drawn from a task distribution [18]–[20] in order to quickly adapt and generalize to new tasks from that distribution with only a few gradient steps, rendering it highly data-efficient for adaptation. We introduce a novel approach to learning-based KKL observer design, harnessing the fundamental principles of MAML to enhance the precision of nonlinear system state estimation through online adaptation to distinct system attributes. The adaptation process exclusively depends on system output measurements to extract inherent system knowledge and dynamically adjust estimation capabilities. This proposed method deviates from the static learning paradigms prevalent in previous methodologies, marking a transition towards enhanced adaptability.

The remainder of the paper is organized as follows. In section II, the theory of KKL observers and the fundamental principles of MAML are provided. In Section III, we present our novel meta-learning-based approach to KKL observer design in detail. We then evaluate the performance of our proposed approach in Section IV, providing empirical results and analysis. Finally, Section V offers our concluding remarks.

Notation

The set of real numbers is denoted by $\mathbb{R}$, $\mathbb{R}^m$ represents the m-dimensional Euclidean space, $\mathbb{R}_{\geq 0}$ the set of nonnegative real numbers. $\mathbb{N}$ respectively denotes the set of natural numbers. The Euclidean norm of the vector $x \in \mathbb{R}^m$ is denoted by $||x||$. The matrix exponential of matrix A is defined as $e^A = \sum_{k=0}^{\infty} \frac{1}{k!} A^k$. Given a finite set S, its cardinality is denoted by $|S|$. The notions 1_N, I_N, and $\mathrm{diag}(1, 2, ..., N)$ respectively denote the N-dimensional vector with all entries equal to one, the $N \times N$ dimensional identity matrix, and the diagonal matrix with entries $1, 2, ..., N$. $[a, b]^m \subset \mathbb{R}^m$ denotes the m-ary Cartesian power of the finite interval $[a, b] \subset \mathbb{R}$.

II. BACKGROUND

A. KKL Observers

We consider a dynamical, autonomous, nonlinear system

$$\begin{cases} \dot{x}(t) & = f\left(x(t)\right) \\ y(t) & = h\left(x(t)\right) \end{cases} \tag{1}$$

where $x \in \mathcal{X} \subset \mathbb{R}^{d_x}$ denotes the nonlinear system state in state space $\mathcal{X}$, and $y \in \mathcal{Y} \subset \mathbb{R}^{d_y}$ the system output. $f : \mathcal{X} \mapsto \mathbb{R}^{d_x}$ and $h : \mathcal{X} \mapsto \mathcal{Y}$ are smooth functions that articulate the nonlinear system dynamics and measurable outputs, respectively. As elucidated in [6]–[8], the formulation of a KKL observer entails the identification of an injective[1] transformation map, denoted by $F : \mathcal{X} \mapsto \mathcal{Z}$, that serves the purpose of converting the dynamics of the nonlinear system (1) into a linear dynamical system described by

$$\dot{z}(t) = Az(t) + By(t), \tag{2}$$

[1] $\forall x_1, x_2 \in \mathcal{X} : F(x_1) = F(x_2) \implies x_1 = x_2$.

where $z(t) \in \mathcal{Z}$ is the linear system state in state space $\mathcal{Z} \subset \mathbb{R}^{d_z}$. Hence, the relationship between the linear state $z(t)$ and the nonlinear state $x(t)$ is expressed as $z(t) = F(x(t))$. It is imperative that the transformation map F satisfies the following partial differential equation:

$$\frac{\partial}{\partial x}\left(F\left(x(t)\right)\right) f\left(x(t)\right) = AF\left(x(t)\right) + Bh\left(x(t)\right). \tag{3}$$

Since the transformation map F is injective, the KKL observer for the nonlinear system (1) is instantiated through the application of the inverse transformation map $F^{-1} : \mathcal{Z} \mapsto \mathcal{X}$ to the linear observer state $z(t)$ and is defined as

$$\begin{cases} \dot{z}(t) & = Az(t) + By(t) \\ x(t) & = F^{-1}\left(z(t)\right). \end{cases} \tag{4}$$

This formulation incorporates the appropriate initialization of the linear observer state with $z(0) = F(x(0))$. In the context of learning-based KKL observers, we approximate F and F^{-1} as depicted by [7] and denote those as $\hat{F}$ and $\hat{F}^{-1}$. In order to ensure the existence of such transformation maps, we make the following assumptions:

Assumption 1: Let the state trajectory of (1) with the initial state $x(0) = x_0$ be expressed by $x(t; x_0)$. There exists a compact set $\mathcal{X} \subset \mathbb{R}^{d_x}$ such that $\forall x_0 \in \mathcal{X}, t \in \mathbb{R}_{\geq 0} : x(t; x_0) \in \mathcal{X}$.

Assumption 2: There exists an open bounded set $\mathcal{S} \supset \mathcal{X}$ wherein (1) is backward $\mathcal{S}$-distinguishable[2].

Previous studies such as [2], [7], [17], [21] emphasize that a uniformly injective[3] map F satisfying (3) exists if (A, B) is controllable, A is Hurwitz, and Assumptions 1 and 2 hold. For formal proof and a more comprehensive context, we direct readers to the cited works.

B. Model-Agnostic Meta-Learning (MAML)

The concept of MAML [18] revolves around training a model on a distribution of tasks, enabling it to rapidly adapt to new tasks drawn from the same distribution. The central notion is to acquire transferable intrinsic features across the given task distribution.

Consider a task T defined as a tuple $T = (u, o)$, where $u \in \mathcal{U}$ denotes the input belonging to the input space $\mathcal{U}$ and $o \in \mathcal{O}$ corresponds to the desired output within the output space $\mathcal{O}$. In the context of meta-learning, we consider a model $\hat{g}_\theta \in \mathcal{G}$ parameterized by θ. Here, $\mathcal{G} \subset \{\mathcal{U} \mapsto \mathcal{O}\}$ denotes the function space containing $\hat{g}_\theta$, which maps inputs to outputs. Additionally, a loss function $L : \mathcal{O} \times \mathcal{U} \times \mathcal{G} \mapsto \mathbb{R}$ is employed. Tasks are sampled from a given task distribution $\mathcal{T}$, denoted as $T \sim \mathcal{T}$. Throughout this distribution, the tasks vary, exposing the model to different inputs and corresponding desired outputs. During the meta-training process, the model

[2] Given an open set $\mathcal{S} \supset \mathcal{X}$, (1) is said to be *backward $\mathcal{S}$-distinguishable* on $\mathcal{X}$ if for every pair of distinct initial conditions $x_{0,1}, x_{0,2} \in \mathcal{X}$, there exists $\tau < 0$ such that $x(t; x_{0,1}), x(t; x_{0,2}) \in \mathcal{S}$ are well-defined for $t \in [\tau, 0]$, and $h(x(\tau; x_{0,1})) \neq h(x(\tau; x_{0,2}))$ [17].

[3] A function $\rho : \mathbb{R}_{\geq 0} \mapsto \mathbb{R}_{\geq 0}$ is said to be of class $\mathcal{K}$ if it is continuous, zero at zero, and strictly increasing. The map F is said to be *uniformly injective* if there exists a class $\mathcal{K}$ function ρ such that $\forall x_1, x_2 \in \mathcal{X} : ||x_1 - x_2|| \leq \rho(||F(x_1) - F(x_2)||)$ [17].

parameters evolve to enable adaptation to specific tasks T using only a limited number of samples and gradient updates.

MAML accomplishes this behavior through a training algorithm characterized by a nested loop. At the beginning of each outer loop iteration, a meta-batch of tasks $\{T_i | T_i \in \mathcal{T}, i = 1, 2, ..., N_{\text{batch,meta}}\}$ is sampled. Subsequently, an inner loop is entered, where the task-specific model parameters θ_i are computed by using *adaptation data points* $(u_{i,\text{a}}, o_{i,\text{a}})$ from each task T_i with the update rule

$$\theta_i = \theta - \alpha \nabla_\theta L(o_{i,\text{a}}, u_{i,\text{a}}, \hat{g}_\theta) \qquad (5)$$

and adaptation learning rate α.

Upon completion of the inner loop updates, a meta-update for the model parameters θ is executed. This update entails the accumulation of losses computed based on a set of predictions for *query data points* $(u_{i,\text{q}}, o_{i,\text{q}})$ which are generated using the updated, task-specific parameter sets θ_i obtained from the inner loop. Mathematically, this update can be expressed as

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{T_i \sim \mathcal{T}} L(o_{i,\text{q}}, u_{i,\text{q}}, \hat{g}_{\theta_i}) \qquad (6)$$

with meta learning rate β. The computation of gradients in (6) encompasses second-order gradients through the inner loop gradients. This results in an optimization process where the parameters in the outer loop are adjusted in a manner aimed at minimizing the loss function through consideration of the inner loop parameter updates, which reflect the task-specific adaptation process.

C. Problem Statement

In this paper, our objective is to find a more general approximation of transformation maps F and F^{-1} given in (3) and (4) through $\hat{F}_\theta$ and $\hat{F}_\eta^{-1}$, realized as feedforward neural network models with learnable parameters θ and η. Given variations in system attributes, such as the initial nonlinear system state $x(0)$ or the intrinsic parameterization of the nonlinear system function $f(\cdot)$, we define a distribution of learning tasks, $\mathcal{T}$, with tasks arising from notable differences in the trajectories $x(t)$ and $z(t)$ of systems (1) and (2) over time t. In this case, employing a meta-learning approach on the task distribution $\mathcal{T}$ is expected to allow for both, better generalization across $\mathcal{T}$, and to adapt to a single task $T \sim \mathcal{T}$ as it is being observed.

III. METHODOLOGY

A. Task-Specific Data Generation

For each task $T \sim \mathcal{T}$, we construct a dataset $\mathcal{D}_T \subset \mathcal{X} \times \mathcal{Z}$ which contains pairs of data samples, denoted as $(x(t; T), z(t; T))$. Here, $x(t; T)$ represents the state trajectory of the nonlinear system related to task T, while $z(t; T)$ corresponds to the state of the linear system, respectively. This synthetic approach of data generation, in line with methodologies presented in [10], [17], employs the RK4 method [22], [23] (or another suitable solver) to simultaneously forward simulate both systems (1) and (2) for a given

task T. The resulting dataset can be expressed as

$$\mathcal{D}_T = \{(x(t; T), z(t; T)) \, | \, t = k \Delta t, k = 0, 1, ..., N\} \qquad (7)$$

where Δt is the time step size used in the solver and $N \in \mathbb{N}$.

Given the analytic solution to the system of linear differential equations in (2) [24]

$$z(t) = e^{At} z(0) + \int_0^t e^{A(t-\tau)} B y(\tau) \mathrm{d}\tau, \qquad (8)$$

the effect of any initial state $z(0) \neq F(x(0))$ would eventually diminish due to the Hurwitz property of A. However, an arbitrary initialization for training data generation might introduce regression errors, particularly for simulated data samples near $t = 0$, depending on the choice of the matrix A. To mitigate such errors resulting from inaccurate training data, we adopt a backward sampling method for (1) and (2). This strategy, as described in prior works such as [17], [25], allows us to obtain the precise initial state $z(0) = F(x(0))$ at which the linear system has reached the steady state:

1) Given some small $\epsilon > 0$, the system matrix A, its eigendecomposition $A = V \Lambda V^{-1}$, and minimum eigenvalue $\lambda_{A,\min}$, compute the time $\tau < 0$ such that $||e^{A(-\tau)} z(\tau)|| < \epsilon$ using

$$\tau \leq \frac{1}{\lambda_{A,\min}} \ln \left(\frac{\epsilon}{\text{cond}(V) \|z(\tau)\|} \right). \qquad (9)$$

2) Simulate $x(t)$ in backward time for $t \in [\tau, 0]$ given $x(0)$ and compute $y(t) = h(x(t))$.

3) Simulate $z(t)$ in forward time for $t \in [\tau, 0]$ given $z(\tau)$ and $y(t)$ from the previous step to obtain $z(0) = F(x(0))$.

B. Forward Computation During Training

During the training process, we employ two distinct modes of forward computation. These modes can be categorized as either sequential or parallel in their application of the map approximations $\hat{F}_\theta$ and $\hat{F}_\eta^{-1}$.

The sequential method reconstructs the nonlinear system state approximation, $\hat{x}(t)$, using the linear system state approximation, $\hat{z}(t)$, as the basis for computation. This is achieved through

$$\hat{z}(t) = \hat{F}_\theta(x(t)), \quad \hat{x}(t) = \hat{F}_\eta^{-1}\left(\hat{F}_\theta(x(t))\right). \qquad (10)$$

In contrast, the parallel method avoids the composition of both map approximations and directly utilizes labels for the state of the linear observer system $z(t)$ to compute $\hat{x}(t)$. The forward computation in this mode can be expressed as

$$\hat{z}(t) = \hat{F}_\theta(x(t)), \quad \hat{x}(t) = \hat{F}_\eta^{-1}(z(t)). \qquad (11)$$

The sequential approach, depicted in Figure 1 (1), applies $\hat{F}_\eta^{-1}$ to the linear state estimate $\hat{z}(t)$, aiming to provide a direct inverse to the approximation $\hat{F}_\theta$. Conversely, the parallel approach, illustrated in Figure 1 (2), applies $\hat{F}_\eta^{-1}$ independently from $\hat{F}_\theta$ to directly invert F.
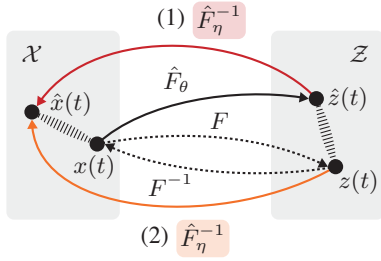
Fig. 1: Modes of forward computation during training, sequential (1) and parallel (2).

C. Mixed-Task Learning Methods

We consider a set of training tasks $\mathcal{T}_{\text{train}}$ sampled from the distribution $\mathcal{T}$ with task-specific datasets $\mathcal{D}_T$ for $T \in \mathcal{T}_{\text{train}}$ and a joint dataset $\mathcal{D} = \cup_{T \in \mathcal{T}_{\text{train}}} \mathcal{D}_T$, which comprises mixed data points corresponding to different tasks in $\mathcal{T}_{\text{train}}$. We address the supervised regression problem to learn $\hat{F}_\theta$ and $\hat{F}_\eta^{-1}$ using (11) on mixed data points from $\mathcal{D}$ [10]. This approach, termed *parallel mixed-task learning*, involves the loss functions

$$L_z\left(z(t), x(t), \hat{F}_\theta\right) = \left\| z(t) - \hat{F}_\theta\left(x(t)\right) \right\|^2$$
$$L_x\left(x(t), z(t), \hat{F}_\eta^{-1}\right) = \left\| x(t) - \hat{F}_\eta^{-1}\left(z(t)\right) \right\|^2, \tag{12}$$

which are independently addressed with the optimization problems

$$\min_\theta L_z\left(z(t), x(t), \hat{F}_\theta\right) \tag{13}$$

and

$$\min_\eta L_x\left(x(t), z(t), \hat{F}_\eta^{-1}\right). \tag{14}$$

Similarly, following the methodology proposed in [17] based on the references therein, we denote the same supervised regression problem when using the sequential forward computation of (10). We refer to this approach as *sequential mixed-task learning*. Induced by the composition of both maps, the loss function L_x in this case becomes

$$L_x\left(x(t), x(t), \hat{F}_\eta^{-1} \circ \hat{F}_\theta\right) = \left\| x(t) - \hat{F}_\eta^{-1}\left(\hat{F}_\theta\left(x(t)\right)\right) \right\|^2, \tag{15}$$

which leads to the optimization problem for θ depending on both L_x and L_z with

$$\min_\theta \left(L_x\left(x(t), x(t), \hat{F}_\eta^{-1} \circ \hat{F}_\theta\right) + L_z\left(z(t), x(t), \hat{F}_\theta\right)\right). \tag{16}$$

The optimization problem for η is then given by

$$\min_\eta L_x\left(x(t), x(t), \hat{F}_\eta^{-1} \circ \hat{F}_\theta\right). \tag{17}$$

D. Meta-Learning for System Output Adaptation

We introduce the concept of *meta-learning for system output adaptation* as a complementary approach to parallel mixed-task learning if $h(\cdot)$ in (1) is known or can be approximated through another data-driven regression approach. We employ the mixed learning method for the approximation $\hat{F}_\theta$

Algorithm 1 Meta-Learning for System Output Adaptation

Require: $\mathcal{T}$ (Distribution of learning tasks)
Require: β (Meta learning rate)
1: Initialize model $\hat{F}_\eta^{-1}$ with parameters η
2: Initialize adaptation learning rate α
3: **while** not done **do**
4:      **for** $i \leftarrow 1, N_{\text{batch,meta}}$ **do**
5:          $\eta_i \leftarrow \eta$ (Initialize task-specific parameters)
6:          $T_i \sim \mathcal{T}$ (Sample learning task)
7:          Generate $\mathcal{D}_{T_i}$
8:          **for** $j \leftarrow 1, N_{\text{adapt}}$ **do**
9:              $(x_{i,\text{a}}, z_{i,\text{a}}) \sim \mathcal{D}_{T_i}$ (Sample adaptation data)
10:            $\eta_i \leftarrow \eta_i - \alpha \nabla_{\eta_i} L_y(h(x_{i,\text{a}}), z_{i,\text{a}}, h \circ \hat{F}_{\eta_i}^{-1})$
11:          **end for**
12:          $(x_{i,\text{q}}, z_{i,\text{q}}) \sim \mathcal{D}_{T_i}$ (Sample query data)
13:      **end for**
14:      $\eta \leftarrow \eta - \beta \nabla_\eta \sum_i L_x(x_{i,\text{q}}, z_{i,\text{q}}, \hat{F}_{\eta_i}^{-1})$
15:      $\alpha \leftarrow \alpha - \beta \nabla_\alpha \sum_i L_x(x_{i,\text{q}}, z_{i,\text{q}}, \hat{F}_{\eta_i}^{-1})$
16: **end while**
17: **return** $\hat{F}_\eta^{-1}$

---

and propose Algorithm 1 to learn the inverse map approximation $\hat{F}_\eta^{-1}$. Inspired by the principles of MAML [18], our meta-learning algorithm follows a nested training loop structure, where inner loop iterations involve multiple *adaptation updates*, while outer loop iterations include *meta-updates*. During each outer iteration, a meta batch of $N_{\text{batch,meta}}$ learning tasks T_i is sampled from $\mathcal{T}$. It is important to emphasize that new tasks T_i are sampled from $\mathcal{T}$ during each training iteration. However, to maintain consistency with the previously introduced mixed-task learning algorithms, we continue to refer to this set of tasks over the entire execution of the training algorithm collectively as $\mathcal{T}_{\text{train}}$. Consequently, we ensure that all learning algorithms are applied to the same training data $\mathcal{D}$ to preserve comparability. For each task T_i, the inner loop performs N_{adapt} optimization steps on a local set of model parameters η_i for adaptation data points $(x_{i,\text{a}}, z_{i,\text{a}}) \sim \mathcal{D}_{T_i}$, utilizing $h(\cdot)$ on estimations $\hat{x}(t)$, the system output loss function defined as

$$L_y\left(y(t), z(t), h \circ \hat{F}_\eta^{-1}\right) = \left\| y(t) - h\left(\hat{F}_\eta^{-1}\left(z(t)\right)\right) \right\|^2 \tag{18}$$

and adaptation learning rate α. Subsequently, the task-specific model parameters η_i are evaluated using query data points $(x_{i,\text{q}}, z_{i,\text{q}}) \sim \mathcal{D}_{T_i}$, and the approximation loss for predicting the nonlinear system state is accumulated across all tasks T_i within the meta batch. In the next step, the accumulated loss is used to perform an optimization step in the outer loop, updating the model parameters η based on the meta-learning rate β. In addition to meta-learning the parameters η, the adaptation learning rate α is also introduced as a trainable parameter during the meta-optimization process. This inclusion enhances accuracy for the specified learning problem and the chosen number of adaptation steps,
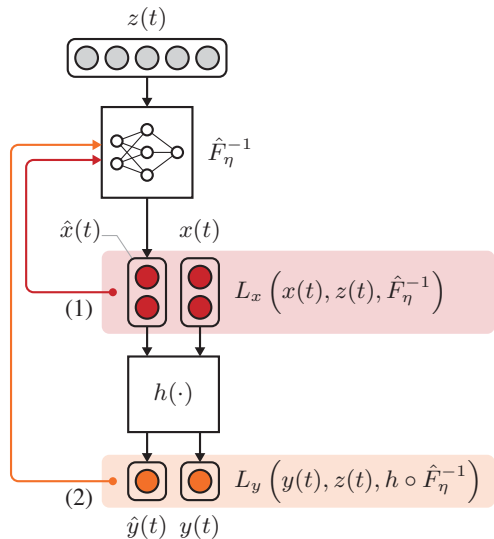
Fig. 2: Computational structure of meta-learning for system output adaptation algorithm. (1) The approximation loss L_x for the entire system state vector is computed for meta-updates only. (2) The approximation loss L_y for the measurable system output contributes to the adaptation updates.

eliminating the need for manual hyperparameter tuning. In essence, the described algorithm seeks to determine a set of model parameters η that enable the model to be updated during the adaptation procedure of the inner loop. The second-order gradients involved in the meta update process adjust η such that task-specific adaptation steps minimize the overall approximation loss L_x. Importantly, as the adaptation updates are computed based on the measurable system output, this approach supports online adaptation if a system output estimate $\hat{y}(t)$ can be derived from the estimated system state $\hat{x}(t)$ following the previously mentioned constraint that $h(\cdot)$ or an approximation is known. The computational structure of meta-learning for system output adaptation is further illustrated in Figure 2.

E. Online Adaptation

To adapt the meta-learned inverse map approximation $\hat{F}_\eta^{-1}$ online to the specific attributes of the observed nonlinear system, we implement the inner loop adaptation updates from Algorithm 1 using data samples drawn from both $y(t)$ and the estimation $\hat{x}(t)$ governed by the unadapted meta-learning-based KKL observer. It is imperative to emphasize the necessity of an initialization period, denoted as t_{init}, which arises from this sampling operation, as depicted in Figure 3.

The minimum duration of t_{init} is determined by the number of data points processed during task-specific adaptation and is expressed as:

$$t_{\mathrm{init}} \geq N_{\mathrm{batch}} \cdot N_{\mathrm{adapt}} \cdot \Delta t, \qquad (19)$$

where N_{batch} represents the data batch size. The sampling period can be modified by either delaying, extending (which includes randomly sampling the required data points from
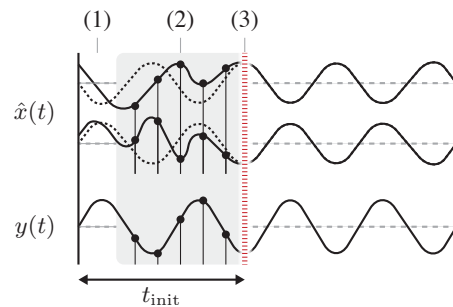

Fig. 3: Online adaptation with illustrated sampling delay (1), sampling and adaptation period for inverse transformation model parameters (2), and onset of high-precision state observation (3).

within), or using a combination of both strategies. Consequently, the impact of the chosen sampling period on the estimation accuracy additionally relies on the linear system dynamics of the KKL observer and the initial state of (2) determined through $\hat{F}_\theta$.

IV. EXPERIMENTS

Building upon experimental setups given in the literature [10], [15]–[17], [25], we assess the accuracy and performance of our proposed meta-learning algorithm for KKL observer by design employing a variation of the Duffing oscillator as nonlinear system, which is defined as follows:

$$\begin{cases} \dot{x}(t) = \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{bmatrix} = \lambda \begin{bmatrix} x_2^3(t) \\ -x_1(t) \end{bmatrix} + w_x(t) \\ y(t) = x_1(t) + w_y(t). \end{cases} \qquad (20)$$

The variable λ denotes an internal parameter that influences the oscillation frequency and the terms $w_x \in \mathbb{R}^{d_x}$ and $w_y \in \mathbb{R}^{d_y}$ account for system and measurement noise, respectively.

The transformation maps $\hat{F}_\theta$ and $\hat{F}_\eta^{-1}$ are approximated through feedforward neural networks with 5 hidden layers of 50 neurons each, ReLU activation functions and normalization layers. This configuration closely follows the framework presented by [17]. We design the linear system (2) of the KKL observer by adopting $A = -\mathrm{diag}(1, 2, ..., d_z)$, $B = 1_{d_z}$ and $d_z = 2d_x + 1 = 5$. It is worth noting that, as pointed out by [17], various adjustments such as an $\mathcal{H}_\infty$ design of matrices A and B, the generation of an arbitrary amount of training data through simulation, or the use of more complex neural network architectures, could potentially enhance accuracy, generalization capability, and robustness against noise. However, we aim to demonstrate the performance of the previously introduced methodologies under conditions where these improvements may be impractical or unattainable. In the following, we evaluate 4 different learning methods[4]: sequential mixed-task learning, physics-informed sequential mixed-task learning as introduced by [17] as *supervised PINN*, parallel mixed-task learning, and

[4]The code implementation for the experiments described in this paper is available at github.com/lukastrm/metakkl.

our proposed method, meta-learning for system output adaptation. We use Adam optimization [26] for the former 3 methods as well as the meta-updates of the meta-learning approach and stochastic gradient-descent optimization for the adaptation updates. The proposed meta-learning approach not only functions as an independent learning algorithm but also exhibits improved performance when the inverse map is pre-trained in a parallel mixed-task learning setting. This strategy is incorporated into all our experiments.

To quantify the accuracy of our estimations for $x(t)$ and $z(t)$, we use normalized estimation errors as follows:

$$e_x^*(t) = \frac{\|x(t) - \hat{x}(t)\|}{\|x(t)\|}, \quad e_z^*(t) = \frac{\|z(t) - \hat{z}(t)\|}{\|z(t)\|}. \quad (21)$$

We furthermore define the normalized estimation error averaged over a set of tasks $\mathcal{T}$ as

$$\bar{e}_{\mathcal{T}}^*(t, e^*, \mathcal{T}) = \frac{1}{|\mathcal{T}|} \sum_{T \in \mathcal{T}} e^*(t; T), \quad (22)$$

and the normalized estimation error for a task $T \in \mathcal{T}$ averaged over time as

$$\bar{e}_t^*(T, e^*) = \frac{1}{N+1} \sum_{k=0}^{N} e^*(k\Delta t; T), \quad (23)$$

where $e^*(t)$ refers to either $e_x^*(t)$ or $e_z^*(t)$, depending on which state space is under evaluation.

We determine these error metrics for a KKL observer with a linear system that has reached the steady state. For online adaptation, we choose the minimum sampling period which also includes drawing the adaptation samples for the meta-trained KKL observer from the non-steady phase of $z(t)$, but also consider other sampling periods in section IV-B.

A. System Parameter Variation

In this experiment, we assess the accuracy of the KKL observer across a distribution of tasks $\mathcal{T}_\lambda$ that incorporate varying values of the parameter λ. The initial nonlinear system state is fixed at $x(0) = [0.5, 0.5]^T$. During training, we employ a small set of 5 distinct parameter values within the range $\lambda \in [1, 5]$ to generate state trajectories. For validation, we extend our analysis to a set of tasks $\mathcal{T}_{\lambda,\text{val}}$, comprising 200 different parameter values for $\lambda \in [1, 5]$ within the training range.

The results, depicted in Figure 4, reveal two significant findings. Firstly, it is evident that the approximation $\hat{F}_\theta$ displays notable inaccuracies when trained on a dataset involving mixed tasks with varying λ. This, in turn, results in substantial estimation errors for the KKL observer with sequential mixed-task learning, primarily due to the inverse map approximation $\hat{F}_\eta^{-1}$ being trained on an imprecise representation of the linear system space. In contrast, parallel mixed-task learning and in particular also our meta-learning approach, utilizing parallel forward computation during training, overcome this limitation. The inverse map $\hat{F}_\eta^{-1}$ is consistently trained on accurately simulated data samples for $z(t)$, thus achieving independence from the accuracy of $\hat{F}_\theta$. Secondly, we emphasize how, despite having a limited
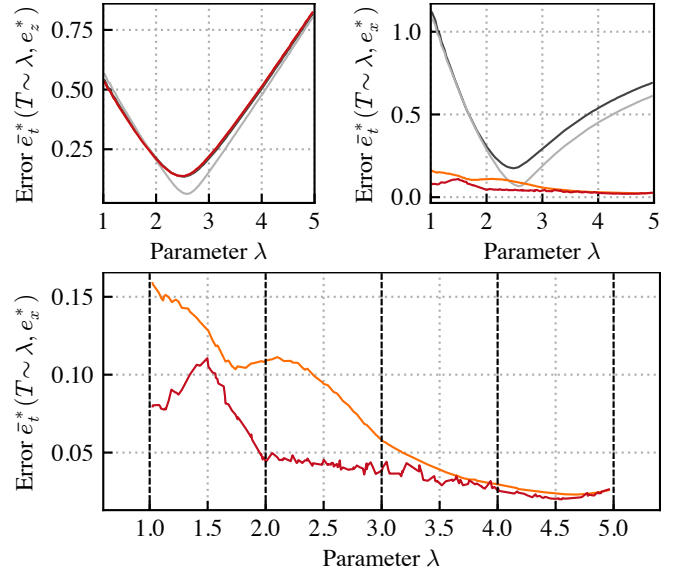

Fig. 4: Simulation estimation error for task distribution $\mathcal{T}_\lambda$ over varying λ for different learning methods: (•) meta-learning with online adaptation, (•) parallel mixed-task learning, (•) sequential mixed-task learning, (•) physics-informed, sequential mixed-task learning. The dashed, black vertical lines in the lower plot indicate the 5 distinct parameter values for λ on which the KKL observer was trained.

amount of training data encompassing state trajectories for only 5 distinct values of λ, our proposed approach demonstrates an improvement in overall validation accuracy through enhanced generalization. Despite the inherent challenge of achieving high accuracy across the entire parameter range due to limited training data, meta-learning and online adaptation leverage the intrinsic system properties acquired from the system output. This allows for more precise estimation of the system state, even for previously unexplored parameter ranges.

B. Initial Nonlinear System State Variation

In this set of experiments, we assess the accuracy of the KKL observer for a distribution of tasks $\mathcal{T}_{x(0)}$ with varying initial nonlinear system states $x(0)$ and fixed parameter value $\lambda = 1$. For training, we draw a set of 50 different initial states $x(0) \in \mathcal{X}_{x(0),\text{train}} \sim [-1, 1]^2$ using latin hypercube sampling [27] to generate state trajectories. For validation, we consider two different sets of tasks:

- $\mathcal{T}_{x(0),\text{val,in}}$ for 50 different initial states within the training range, i.e. $x(0) \in [-1, 1]^2 \setminus \mathcal{X}_{x(0),\text{train}}$
- $\mathcal{T}_{x(0),\text{val,out}}$ for 80 different initial states outside of the training range with $x(0) \in [-2, 2]^2 \setminus [-1, 1]^2$

To evaluate accuracy under the influence of noise, we introduce Gaussian noise components with $w_x(t) \sim \mathcal{N}(0, 0.1)$ and $w_y(t) \sim \mathcal{N}(0, 0.1)$.

Figures 5a and 5b display the estimated nonlinear system state trajectory $\hat{x}(t)$ for one specific $x(0)$ using the meta-learning-based KKL observer with online adaptation, clearly
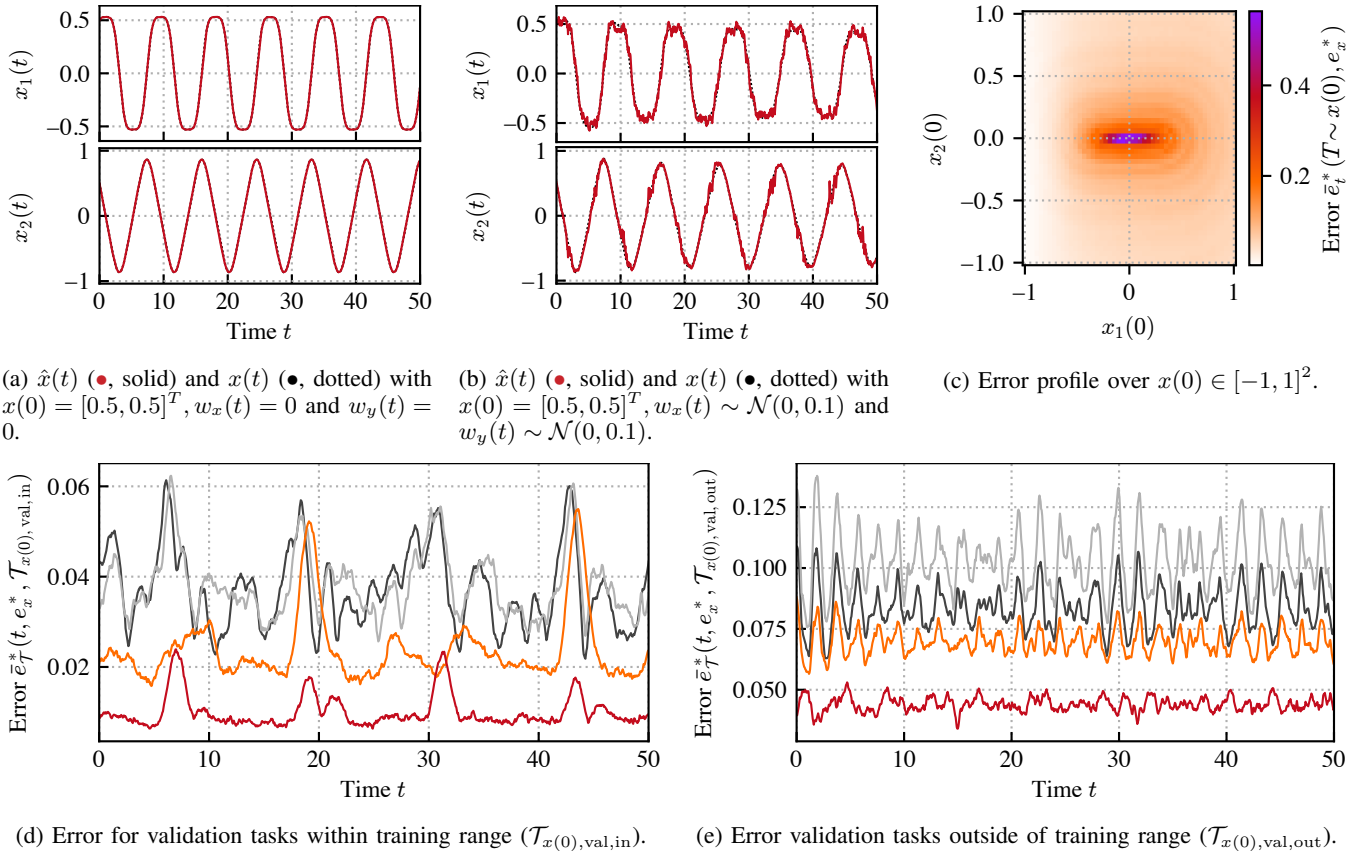
(a) $\hat{x}(t)$ (•, solid) and $x(t)$ (•, dotted) with $x(0) = [0.5, 0.5]^T$, $w_x(t) = 0$ and $w_y(t) = 0$.

(b) $\hat{x}(t)$ (•, solid) and $x(t)$ (•, dotted) with $x(0) = [0.5, 0.5]^T$, $w_x(t) \sim \mathcal{N}(0, 0.1)$ and $w_y(t) \sim \mathcal{N}(0, 0.1)$.

(c) Error profile over $x(0) \in [-1, 1]^2$.

(d) Error for validation tasks within training range ($\mathcal{T}_{x(0), \text{val,in}}$).

(e) Error validation tasks outside of training range ($\mathcal{T}_{x(0), \text{val,out}}$).

Fig. 5: Simulation results for task distribution $\mathcal{T}_{x(0)}$ over varying $x(0)$ for different learning methods: (•) meta-learning with online adaptation, (•) parallel mixed-task learning, (•) sequential mixed-task learning, (•) physics-informed, sequential mixed-task learning.

highlighting its ability to deliver accurate state estimations, even in the presence of noise.

In Figure 5d, we present the error (22) for the validation task set associated with initial system states within the training range. Notably, our proposed methodology consistently achieves the highest accuracy estimations on average when compared to other learning methods. Furthermore, our approach demonstrates strong generalization capabilities, evidenced by the notably low estimation error for initial system states outside of the training range, as portrayed in Figure 5e.

Figure 5c shows the profile of the error (23) across the entire training range. In can be observed that the normalized error is larger for initial system states around $x(0) = [0, 0]^T$, which can be contributed to numerical imprecisions that arise from a wide normalization range in combination with an absolute loss function during training. This effect has previously also been observed in [10].

We further explore the impact of the chosen adaptation sampling period on the precision of the KKL observer. Figure 6 presents the simulation results for the meta-learning-based KKL observer with online adaptation, employing four different variations of the sampling strategy. The results highlight that, in the case of the periodic Duffing oscillator,

opting for the minimum sampling window without delay leads to the least accurate estimation outcomes. As the linear system error, originating from the initialization using $\hat{F}_\theta$, diminishes over time, expanding the sampling window contributes to improved accuracy. This improvement is due to estimations for $\hat{x}(t)$ being derived from progressively less erroneous samples in $\mathcal{Z}$. Notably, the most accurate estimation performance is observed for delayed sampling windows, ensuring that any error stemming from an inaccurate initial state in $\mathcal{Z}$ has dissipated before the initial adaptation samples are drawn.

V. CONCLUSION

In this paper, we have introduced a novel meta-learning-based algorithm to approximate the inverse transformation map, making it adaptable to different system attributes, and conducted a comprehensive comparison of various learning-based KKL observer approaches. We have demonstrated through experiments that our proposed method, integrated with the parallel mixed-task learning algorithm, exhibits heightened estimation accuracy for scenarios both within and outside the training range. This underscores the enhanced generalization capability of meta-learning. Our methodology has also displayed adaptability over varying system param-
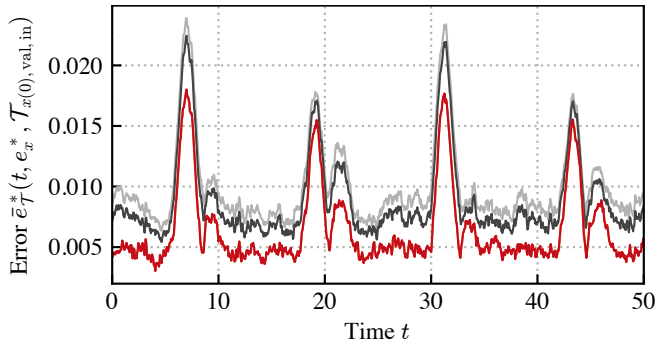
Fig. 6: Simulation results for task distribution $\mathcal{T}_{x(0)}$ over varying $x(0)$ for meta-learned KKL observer with different sampling periods for online adaptation: (•) minimum sampling period without delay, (•) sampling period of size $t = 50$ and random sampling of data points within, (•) minimum sampling period delayed by $-\tau$, (•) sampling period of size $t = 50$ delayed by $-\tau$.

eters, even when only limited task-specific data is available. The integration of meta-learning principles, leveraging the ability to comprehend intrinsic features of the addressed task distribution, has exhibited effectiveness, notably in situations where learning the transformation map F proves challenging within a given setup. This especially outpaces approaches incorporating sequential forward computation during training.

Our future research will focus on integrating various enhancements, such as those outlined by the authors of [28], which could potentially enhance accuracy and foster a more stable learning behavior. Additionally, the necessity of an adaptation sampling period requires careful evaluation. Meta-learning-based KKL observers for systems with dynamically changing attributes may benefit from periodic re-adaptation, but the frequency of attribute changes is constrained by the initialization time of the observer.

REFERENCES

[1] D. G. Luenberger, "Observing the State of a Linear System," *IEEE Transactions on Military Electronics*, vol. 8, no. 2, pp. 74–80, 1964.

[2] P. Bernard, V. Andrieu, and D. Astolfi, "Observer design for continuous-time dynamical systems," *Annual Reviews in Control*, vol. 53, pp. 224–248, 2022.

[3] M. Zeitz, "The extended Luenberger observer for nonlinear systems," *Systems & Control Letters*, vol. 9, no. 2, pp. 149–156, 1987.

[4] K. Reif and R. Unbehauen, "The extended Kalman filter as an exponential observer for nonlinear systems," *IEEE Transactions on Signal processing*, vol. 47, no. 8, pp. 2324–2328, 1999.

[5] M. Boutayeb and D. Aubry, "A strong tracking extended Kalman observer for nonlinear discrete-time systems," *IEEE Transactions on Automatic Control*, vol. 44, no. 8, pp. 1550–1556, 1999.

[6] N. Kazantzis and C. Kravaris, "Nonlinear observer design using Lyapunov's auxiliary theorem," *Systems & Control Letters*, vol. 34, no. 5, pp. 241–247, 1998.

[7] V. Andrieu and L. Praly, "On the Existence of a Kazantzis–Kravaris/Luenberger Observer," *SIAM Journal on Control and Optimization*, vol. 45, no. 2, pp. 432–456, 2006.

[8] V. Andrieu, "Convergence speed of nonlinear Luenberger observers," *SIAM Journal on Control and Optimization*, vol. 52, no. 5, pp. 2831–2856, 2014.

[9] P. Bernard and V. Andrieu, "Luenberger Observers for Nonautonomous Nonlinear Systems," *IEEE Transactions on Automatic Control*, vol. 64, no. 1, pp. 270–281, 2019.

[10] L. d. C. Ramos, F. Di Meglio, V. Morgenthaler, L. F. F. da Silva, and P. Bernard, "Numerical design of Luenberger observers for nonlinear systems," in *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, 2020, pp. 5435–5442.

[11] V. Andrieu and P. Bernard, "Remarks about the numerical inversion of injective nonlinear maps," in *2021 60th IEEE Conference on Decision and Control (CDC)*, 2021, pp. 5428–5434.

[12] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[13] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[14] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[15] J. Peralez and M. Nadri, "Deep learning-based luenberger observer design for discrete-time nonlinear systems," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 4370–4375.

[16] J. Peralez, M. Nadri, and D. Astolfi, "Neural Network-Based KKL observer for nonlinear discrete-time systems," in *2022 IEEE 61st Conference on Decision and Control (CDC)*, 2022, pp. 2105–2110.

[17] M. U. B. Niazi, J. Cao, X. Sun, A. Das, and K. H. Johansson, "Learning-based design of Luenberger observers for autonomous nonlinear systems," in *2023 American Control Conference (ACC)*. IEEE, 2023, pp. 3048–3055.

[18] C. Finn, P. Abbeel, and S. Levine, "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks," in *International conference on machine learning*. PMLR, 2017, pp. 1126–1135.

[19] C. Finn, A. Rajeswaran, S. Kakade, and S. Levine, "Online Meta-Learning," in *International Conference on Machine Learning*. PMLR, 2019, pp. 1920–1930.

[20] A. Fallah, A. Mokhtari, and A. Ozdaglar, "On the Convergence Theory of Gradient-Based Model-Agnostic Meta-Learning Algorithms," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 1082–1092.

[21] L. Brivadis, V. Andrieu, P. Bernard, and U. Serres, "Further remarks on KKL observers," *Systems & Control Letters*, vol. 172, p. 105429, 2023.

[22] C. Runge, "Ueber die numerische Auflösung von Differentialgleichungen." *Mathematische Annalen*, vol. 46, pp. 167–178, 1895.

[23] W. Kutta, *Beitrag zur näherungsweisen Integration totaler Differentialgleichungen*. Teubner, 1901.

[24] C.-T. Chen, *Linear System Theory and Design*. Saunders college publishing, 1984.

[25] M. Buisson-Fenet, L. Bahr, V. Morgenthaler, and F. Di Meglio, "Towards gain tuning for numerical KKL observers," *arXiv preprint arXiv:2204.00318*, 2022.

[26] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015.

[27] J. C. H. Ronald L. Iman and J. E. Campbell, "An Approach to Sensitivity Analysis of Computer Models: Part I — Introduction, Input Variable Selection and Preliminary Variable Assessment," *Journal of Quality Technology*, vol. 13, no. 3, pp. 174–183, 1981.

[28] A. Antoniou, H. Edwards, and A. J. Storkey, "How to train your MAML," *CoRR*, vol. abs/1810.09502, 2018.