2018 European Control Conference (ECC)
June 12-15, 2018. Limassol, Cyprus

Nonlinear Set Membership Regression with Adaptive Hyper-Parameter Estimation for Online Learning and Control*

Jan-Peter Calliess[1], Stephen Roberts[1], Carl Rasmussen[2] and Jan Maciejowski[2]

***Abstract*— Methods known as Lipschitz Interpolation or Nonlinear Set Membership regression have become established tools for nonparametric system-identification and data-based control. They utilise *presupposed* Lipschitz properties to compute inferences over unobserved function values. Unfortunately, they rely on the a priori knowledge of a Lipschitz constant of the underlying target function which serves as a hyper-parameter. We propose a closed-form estimator of the Lipschitz constant that is robust to bounded observational noise in the data. The merger of Lipschitz Interpolation with the new hyper-parameter estimator gives a new nonparametric machine learning method for which we derive online learning convergence guarantees. Furthermore, we apply our learning method to model-reference adaptive control and provide a convergence guarantee on the closed-loop dynamics. In a simulated flight manoeuvre control scenario, we compare the performance of our approach to recently proposed alternative learning-based controllers.**

I. Introduction

Among supervised learning methods, nonparametric regression techniques have attracted much attention due to their great flexibility to learn rich function classes. Among many others, popular approaches include kernel methods such as *Gaussian Processes (GPs)* [17], the *NW-estimator* [16], [22], local methods such as *LOESS* regression [11] as well as *Lipschitz Interpolation (LI)* [21], [23]. In spite a wealth of classic as well as recent work that has shed light on the theoretical and practical properties of these methods, a common limitation remains: typically all results rest on the assumption of the knowledge of a suitable hyper-parameter that encodes a priori knowledge about the underlying learning target. While for some methods, especially for many of the kernel methods with certain choices of kernels, asymptotic consistency guarantees can be given for general classes of target functions, irrespective of the chosen hyper-parameter, in practice, the choice of hyper-parameter markedly impacts the predictive performance of the regression method for finite data sets. In *Lipschitz Interpolation (LI)* or *Nonlinear Set Membership (NSM)* methods [21], [13], [23], the hyper-parameter is a Lipschitz constant of the predictor. If set too low, the class of learnable target functions is too restrictive. If on the other hand the parameter is set too high, the resulting predictor will tend to overfit to noise in the data and might yield poor generalisation performance. Therefore, a common solution is to resort to hyper-parameter optimisation [17], [5]. While often working well in practice, these approaches tend to be too computationally expensive to work with large data and to support online learning and adaptive control. Moreover, to the best of our knowledge, no theoretical insights into the learning-theoretic properties of the inferences with the hyper-parameter optimisers in place exist to date.

For Lipschitz Interpolation (LI), this paper addresses this gap. To this end, we propose a closed-form expression to estimate the Lipschitz constant from the data that is a modification of Strongin's estimator [20]. It has the benefit to support computationally tractable online updates but also offers robustness to (bounded) observational noise. We then propose to utilise the estimates in the LI rule to make predictions of function values at unobserved inputs. This combination of Lipschitz constant estimator and LI yields a new nonparametric regression method which we refer to as Lazily Adaptive Constant Kinky Inference (LACKI). We provide convergence guarantees on the prediction errors in a supervised online learning setting. To illustrate some of the benefits and shortcomings of our approach, we compare LACKI with a selection of established regression methods in learning-based tracking control where it outperforms competing approaches across a range of performance metrics and problem setups. This paper is a short version of a preprint [4]. The longer version includes more detailed proofs and additional guarantees such as sample complexity bounds and worst-case consistency guarantees for offline learning settings.

II. Lipschitz Interpolation with adaptive Lipschitz constant estimates

Setting. Let $\mathcal{X}$ be an input space endowed with (pseudo-) metric $\mathfrak{d} : \mathcal{X}^2 \to \mathbb{R}_{\geq 0}$ and let $\mathcal{Y}$ be an output (vector) space endowed with a translation-invariant pseudo-metric $\mathfrak{d}_{\mathcal{Y}} : \mathcal{Y}^2 \to \mathbb{R}_{\geq 0}$. Let $\mathrm{Lip}(L) = \{\phi : \mathcal{X} \to \mathcal{Y} | \, \mathfrak{d}_{\mathcal{Y}}(\phi(x), \phi(x')) \leq L \, \mathfrak{d}(x, x'), \forall x, x' \in \mathcal{X}\}$ denote the set of Lipschitz continuous functions with Lipschitz constant L. The best Lipschitz constant of a function f is the smallest number L^* such that $f \in \mathrm{Lip}(L^*)$. A function is Lipschitz continuous if it has a finite Lipschitz constant.

Let $f : \mathcal{X} \to \mathcal{Y}$ be a *target* function we desire to learn in a supervised fashion. To this end, we assume that, at time step n, we have access to a *sample* or *data set* $\mathcal{D}_n := \{(s_i, \tilde{f}_i) \, | \, i = 1, \ldots, N_n\}$ containing $N_n \in \mathbb{N}$ (possibly corrupted) sample values $\tilde{f}_i \in \mathcal{Y}$ of *target function* f at sample input $s_i \in \mathcal{X}$. The sampled function values are allowed to have *observational error* given by an error function $\mathfrak{e} : \mathcal{X} \to \mathbb{R}^m_{\geq 0}$ which may model stochastic noise or systematic error. That is, we assume $\mathfrak{d}_{\mathcal{Y}}(\tilde{f}_i, f(s_i)) \leq \mathfrak{e}(s_i)$.

Part of this work was supported via EPSRC NMZR/031 RG64733.

[1]OMI, Dept. of Engineering Science, University of Oxford, UK.
[2]Engineering Department, University of Cambridge, UK.

For convenience, we may also write $\mathcal{D}_n = (G_n, \mathcal{Y}_n)$ where $G_n = \{s_i | i = 1, ..., N_n\} \subset \mathcal{X}$ is the collection (or *grid*) of sample inputs and $\mathcal{Y}_n = \{\tilde{f}_i | i = 1, ..., N_n\} \subset \mathcal{Y}$ is the pertaining sequence of observed function values. It is our aim to learn target function f in the sense that we utilise the available data $\mathcal{D}_n$ to infer *predictions* $\hat{\mathfrak{f}}_{\mathfrak{n}}(x)$ of $f(x)$ at unobserved *query inputs* $x \notin G_n$. In our context, the evaluation of $\hat{\mathfrak{f}}_n$ is what we refer to as *(inductive) inference* or *prediction* and $\hat{\mathfrak{f}}_{\mathfrak{n}}$ is referred to as the *predictor*.

Learning rule. We will now rehearse a simplified version of *Kinky Inference (KI)* [6] – a class of nonparametric learning rules that encompasses a host of other methods such as NSM methods [13] and standard Lipschitz Interpolation [21], [3], [23]. As a special case, we will then define our proposed method that incorporates an adaptive estimator of the Lipschitz constant of the target.

Definition II.1 (Kinky inference (KI) rule -simplified)**.** Given access to a sample set $\mathcal{D}_n$ and an input space pseudo-metric $\tilde{\mathfrak{d}}(\cdot, \cdot; \theta(n)) : \mathcal{X}^2 \to \mathbb{R}$ parameterised by $\theta(n)$, we define the KI predictor by $\hat{\mathfrak{f}}_{\mathfrak{n}}\big(\cdot; \theta(n), \mathcal{D}_n\big) : \mathcal{X} \to \mathcal{Y}$ to perform inference over function values as per:

$$\hat{\mathfrak{f}}_{\mathfrak{n}}\big(x; \theta(n), \mathcal{D}_n\big) := \frac{1}{2}\mathfrak{u}_n\big(x; \theta(n)\big) + \frac{1}{2}\mathfrak{l}_n\big(x; \theta(n)\big). \tag{1}$$

Here, $\mathfrak{u}_n\big(\cdot; \theta(n)\big), \mathfrak{l}_n\big(\cdot; \theta(n)\big) : \mathcal{X} \to \mathbb{R}^m$ are defined by $\mathfrak{u}_n\big(x; \theta(n)\big) := \min_{i=1,...,N_n} \tilde{f}_i + \tilde{\mathfrak{d}}(x, s_i; \theta(n))$ and $\mathfrak{l}_n\big(x; \theta(n)\big) := \max_{i=1,...,N_n} \tilde{f}_i - \tilde{\mathfrak{d}}(x, s_i; \theta(n))$, respectively.

The computational effort for making a prediction is in $\mathcal{O}(N_n\, M)$ where M is the effort for evaluating the pseudo-metric. However, it is possible to apply (generalised) nearest-neighbour techniques to reduce this effort to expected logarithmic growth in the number of sample points [3], [6].

A special case arises for the choice of $\tilde{\mathfrak{d}}(x, y; \theta(n)) = L(n)\, \|x - y\|$ which is referred to as *Lipschitz Interpolation* [3] or as *Nonlinear Set Interpolation* [13]. Here the parameter $\theta(n) = L(n)$ is the supposed Lipschitz constant of the target. We will see below that the predictor $\hat{\mathfrak{f}}_{\mathfrak{n}}(\cdot; L(n), \mathcal{D}_n)$ is indeed Lipschitz continuous with Lipschitz constant $L(n)$. Typically, this constant is assumed to be either known a priori or estimated from the data, e.g. [20], [13], [5]. Unfortunately, little is understood about the effects of these previously proposed parameter estimation techniques on the predictor's performance and about the impact of observational noise.

We will now define our *Lazily Adapted Kinky Inference (LACKI)* learning rule that will allow us to provide online learning guarantees even when parameters are estimated from the data online.

For notational convenience, for two sets $S, S' \subset \mathcal{X}$ of inputs we define $U(S, S') := \{(s, s') \in S \times S' | \, \mathfrak{d}(s, s') > 0\}$ and let $U_n := U(G_n, G_n)$ be the set of all pairs grid inputs deemed disparate under the pseudo-metric $\mathfrak{d}$.

Definition II.2 (LACKI rule)**.** The Lazily Adapted Lipschitz Constant Kinky Inference (LACKI) rule computes a KI predictor $\hat{\mathfrak{f}}_{\mathfrak{n}}$ as per Defn. 1, but where $\tilde{\mathfrak{d}}(x, x'; L(n)) = L(n)\, \mathfrak{d}(x, x')$ and where we set

$$L(n) := \max\Big\{0, \max_{(s,s') \in U_n} \frac{\mathfrak{d}_{\mathcal{Y}}(\tilde{f}(s), \tilde{f}(s')) - \lambda}{\mathfrak{d}(s, s')}\Big\}. \tag{2}$$

Note, $\lambda \geq 0$ is a design parameter. When we set $\lambda = 2\bar{\mathfrak{e}}$ (where $\bar{\mathfrak{e}} \in \mathbb{R}_{\geq 0}$ is the lowest upper bound on the level of observational noise, i.e. $\mathfrak{d}_{\mathcal{Y}}(0, \mathfrak{e}(x)) \leq \bar{\mathfrak{e}}, \forall x$), it is easy to see that $L(n)$ is bounded for Lipschitz continuous target f. Being a Lipschitz constant of the predictor boundedness of $L(n)$ can cause the predictor to smooth out i.i.d. observational noise. And, similar to other non-parametric regression methods this noise hyper-parameter does impact the generalisation method's performance (for an illustration, cf. Fig. 1, LACKI vs LACKI2). In general, as we make no distributional assumptions about the observational noise (in particular it could be systematic error), our convergence guarantees we derive below will generally have to depend on it. Next, consider an online learning situation where the available data grows incrementally such that $G_{n+1} = G_n \cup \{s_{n+1}\}, \forall n$. We can define an incremental update rule recursively as follows: $L(n+1) := \max\Big\{L(n), \max_{(s,s') \in U(G_n, \{s_{n+1}\})} \frac{\mathfrak{d}_{\mathcal{Y}}\big(\tilde{f}(s), \tilde{f}(s')\big) - \lambda}{\mathfrak{d}(s, s')}\Big\}$ for $n \in \mathbb{N}$ and where $L(0) := 0$. The effort of computing $L(n+1)$ in time step $n+1$ based on the newly arrived sample point and the previous Lipschitz constant estimate $L(n)$ is in $\mathcal{O}\big(N_n\big)$.

A. Some properties

In preparation of subsequent parts of this paper, we will conclude this section by deriving Lipschitz continuity as well as sample consistency properties of our LACKI inference rule. For simplicity, we will henceforth assume canonical output space and norm-induced metrics, i.e. $\mathcal{Y} = \mathbb{R}^m$ and $\mathfrak{d}_{\mathcal{Y}}\big(y, y'\big) = \|y - y'\|_{\infty}$.

Lemma II.3 (Lipschitz regularity of LACKI)**.** *The predictors* $\hat{\mathfrak{f}}_{\mathfrak{n}}$ *are Lipschitz continuous* $(n \in \mathbb{N})$ *with constant* $L(n)$*. That is,*

$$\Big\| \hat{\mathfrak{f}}_{\mathfrak{n}}(x) - \hat{\mathfrak{f}}_{\mathfrak{n}}(x') \Big\|_{\infty} \leq L(n)\ \mathfrak{d}(x, x'), \forall n \in \mathbb{N}, x, x' \in \mathcal{X}.$$

Proof. (sketch) It is easy to show that the one-dimensional mappings of the form $x \mapsto \ell\, \mathfrak{d}_{\mathcal{X}}\big(x, x'\big)$ are $\ell-$ Hölder continuous for any choices of ℓ and inputs x'. Furthermore, taking point-wise $\max, \min$ as well as averages of Lipschitz continuous functions is known to not change their Lipschitz properties (e.g. [6]). Therefore, all output-component predictors $\hat{\mathfrak{f}}_{n,j}$ $(j = 1, ..., m)$ are $L(n)$- Lipschitz. □

Lemma II.4 (Sample-consistency of LACKI)**.** *If for each output dimension* $j \in \{1, ..., d\}$ *and some* $\lambda \geq 0$ *we have* $L(n) \geq \max_{(s,s') \in U_n} \frac{\big|\tilde{f}_j(s) - \tilde{f}_j(s')\big| - \lambda}{\mathfrak{d}(s, s')}$ *then the LACKI rule is sample-consistent (up to* $\frac{\lambda}{2}$*). That is,* $\forall q \in \{1, \ldots, N_n\}$: $\hat{\mathfrak{f}}_{\mathfrak{n}}(s_q) \in \mathfrak{B}_{\frac{\lambda}{2}}\big(\tilde{f}_q\big)$ *where* $\mathfrak{B}_{\frac{\lambda}{2}}\big(\tilde{f}_q\big) = \{x \in \mathcal{Y} | \, \Big\|x - \tilde{f}_q\Big\|_{\infty} \leq \frac{\lambda}{2}\}$ *denotes the* $\frac{\lambda}{2}$*-ball around the observation* $\tilde{f}_q$*.*

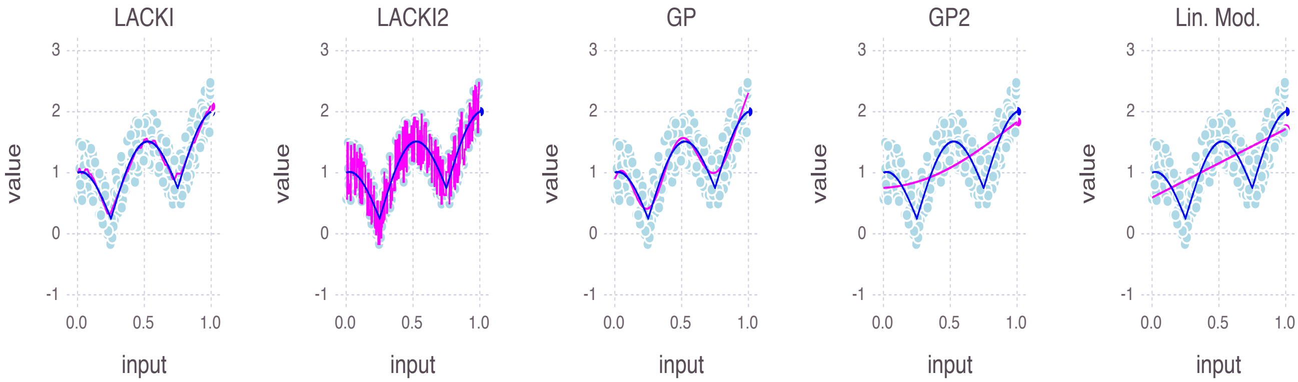

Fig. 1. The predictors of several regression methods for a target function $f : x \mapsto |\cos(2\pi x)| + x$. The $N_n = 500$ observations (light blue dots) in the sample were perturbed by uniform noise drawn i.i.d. from the interval $[-.5, .5]$. The target function is plotted in dark blue and the predictions of the trained models are plotted in magenta. From left to right: LACKI: Our LACKI method with correctly set noise parameter $\bar{\mathfrak{e}}$. LACKI2: LACKI with falsely set noise parameter $\bar{\mathfrak{e}} = 0$. GP: Mean of a Gaussian process regressor with manually tweaked kernel hyper-parameter falsely assuming no observational noise. GP2: automatically optimised kernel hyper-parameter assuming correct noise variance causing to model over-smoothing. Lin. Mod.: LMS regression of a linear model.

Thus, we also have

$$\left\| f(s_q) - \hat{\mathfrak{f}}_{\mathfrak{n}}(s_q) \right\|_\infty \leq \frac{\lambda}{2} + \|\mathfrak{e}(s_q)\|_\infty \leq \frac{\lambda}{2} + \bar{\mathfrak{e}}.$$

Proof. Remember, our output-space metric is given by $\mathfrak{d}_{\mathcal{Y}}(y, y') = \|y - y'\|_\infty$. For ease of notation, we will confine our proof to the case of one-dimensional outputs ($d = 1$). The multi-dimensional case follows trivially from the one-dimensional result by applying it to each output component function. Let $n \in \mathbb{N}$ be fixed and, for ease of notation, write $L := L(n)$. Let $j, k \in \{1, \ldots, N_n\}$ such that $j \in \operatorname{argmin}_i \tilde{f}_i + L\,\mathfrak{d}(s_i, s_q)$ and $k \in \operatorname{argmax}_i \tilde{f}_i - L\,\mathfrak{d}(s_i, s_q)$. By definition of $\hat{\mathfrak{f}}_{\mathfrak{n}}$ we have:

$$\hat{\mathfrak{f}}_{\mathfrak{n}}(s_q) = \frac{1}{2}\Big(\underbrace{\tilde{f}_j + L\,\mathfrak{d}(s_j, s_q)}_{:=B}\Big) + \frac{1}{2}\Big(\underbrace{\tilde{f}_k - L\,\mathfrak{d}(s_k, s_q)}_{=:A}\Big). \quad (3)$$

(i) Firstly, we show $\underline{A \in [\tilde{f}_q, \tilde{f}_q + \lambda]}$: If $k = q$, this holds trivially true since then $A = \tilde{f}_q$. So, assume $k \neq q$. We have $\tilde{f}_k \geq \tilde{f}_k - L\,\mathfrak{d}(s_k, s_q) \geq \tilde{f}_q - L\,\mathfrak{d}(s_q, s_q) = \tilde{f}_q$ where the second inequality holds due to $k \in \arg\max_i \tilde{f}_i - L\,\mathfrak{d}(s_i, s_q)$. That is,

$$A = \tilde{f}_k - L\,\mathfrak{d}(s_k, s_q) \geq \tilde{f}_q. \quad (4)$$

On the other hand, since $L \geq \max_{(s,s') \in U_n} \frac{\left|\tilde{f}(s) - \tilde{f}(s')\right| - \lambda}{\mathfrak{d}(s,s')}$ we have in particular: $L \geq \frac{\left|\tilde{f}_k - \tilde{f}_q\right| - \lambda}{\mathfrak{d}(s_k, s_q)}$. Thus, $L\,\mathfrak{d}(s_k, s_q) + \lambda \geq \left|\tilde{f}_k - \tilde{f}_q\right| = \tilde{f}_k - \tilde{f}_q$. Hence, $\tilde{f}_q + \lambda \geq \tilde{f}_k - L\,\mathfrak{d}(s_k, s_q) = A$. Together with (4) we have shown $A \in [\tilde{f}_q, \tilde{f}_q + \lambda]$.

(ii) The proof of $\underline{B \in [\tilde{f}_q - \lambda, \tilde{f}_q]}$ is completely analogous to that of (i) and hence, is omitted.

(iii) Together, the statements in (i) and (ii) entail $\hat{\mathfrak{f}}_{\mathfrak{n}}(s_q) = \frac{1}{2}A + \frac{1}{2}B \in [\tilde{f}_q - \frac{\lambda}{2}, \tilde{f}_q + \frac{\lambda}{2}]$.

Hence, $\mathfrak{d}_{\mathcal{Y}}\big(\hat{\mathfrak{f}}_{\mathfrak{n}}(s_q), \tilde{f}(s_q)\big) \leq \frac{\lambda}{2}$.

Moreover, for any sample input s_q we have $\hat{\mathfrak{f}}_{\mathfrak{n}}(s_q) = f(s_q) + \phi_q + \psi_q$ with $\mathfrak{d}_{\mathcal{Y}}(0, \psi_q) \leq \frac{\lambda}{2}, \mathfrak{d}_{\mathcal{Y}}(0, \phi_q) \leq \mathfrak{d}_{\mathcal{Y}}\big(0, \mathfrak{e}(s_q)\big) \leq \bar{\mathfrak{e}}$. Our output-space metric is translation-invariant and hence, $\mathfrak{d}_{\mathcal{Y}}\big(f(s_q), \hat{\mathfrak{f}}_{\mathfrak{n}}(s_q)\big) = \mathfrak{d}_{\mathcal{Y}}\big(0, \hat{\mathfrak{f}}_{\mathfrak{n}}(s_q) - f(s_q)\big) = \mathfrak{d}_{\mathcal{Y}}\big(0, \phi_q + \psi_q\big) \leq \frac{\lambda}{2} + \mathfrak{d}_{\mathcal{Y}}\big(0, \mathfrak{e}(s_q)\big) \leq \frac{\lambda}{2} + \bar{\mathfrak{e}}$. □

III. Online Learning Guarantees

In the long version of the paper [4], we prove consistency results. That is, asymptotics and sample complexity bounds for the case where the data becomes dense in the domain (with high probability). In this shorter version of the paper however, we will instead consider an online learning setting where we incrementally get to observe samples along the trajectory of inputs $\Big(x_n\Big)_{n \in \mathbb{N}}$ and are interested in the long-term one-step-lookahead prediction errors on this trajectory, irrespective of distributional assumptions. That is, we are interested in the evolution of prediction errors $\mathfrak{d}_{\mathcal{Y}}\big(\hat{\mathfrak{f}}_{\mathfrak{n}}(x_n), f(x_n)\big)$ where the predictor $\hat{\mathfrak{f}}_{\mathfrak{n}}(\cdot)$ is based on $\mathcal{D}_n = \mathcal{D}_{n-1} \cup \big\{\big(x_{n-1}, \tilde{f}(x_{n-1})\big)\big\}, \forall n > 1$.

We will show that this error trajectory vanishes (up to observational errors), provided that the input sequence $\Big(x_n\Big)_{n \in \mathbb{N}}$ is bounded.

Lemma III.1. *Assume we are given a trajectory* $\Big(x_n\Big)_{n \in \mathbb{N}}$ *of inputs with $x_n \in \mathcal{X}$ where input space $\mathcal{X}$ can be endowed with a shift-invariant measure. Furthermore, assume the sequence is bounded, i.e. $\mathfrak{d}_{\mathcal{X}}(x_n, 0) \leq \beta$ for some $\beta \in \mathbb{R}_+$ and all $n \in \mathbb{N}$. Finally, assume the inputs of the available data coincide with the complete history of past inputs, i.e. $G_n = \{x_i | i \in \mathbb{N}, i < n\}$. Then we have:*

$$\mathit{dist}(G_n, x_n) = \min\{\, \mathfrak{d}_{\mathcal{X}}(g, x_n) |\, g \in G_n\} \overset{n \to \infty}{\longrightarrow} 0.$$

Proof. (sketch) If the distances were not to converge, there existed an infinite number of disjoint balls around the input points that summed up to infinite volume. This however, would be a contradiction to the presupposed boundedness of the sequence. For the full proof, refer to [4]. □

We are now in a position to prove that the prediction errors will vanish the longer the (online) learning proceeds:

Theorem III.2. *Assume that, for some $q \geq 0$, we chose $\lambda = 2\bar{\mathfrak{e}} + q$ in our LACKI prediction rule. And, let f be Lipschitz continuous up to some error level $\bar{E}_h$. That is, $f = \phi + \psi$ with $\phi \in Lip(L^*)$ and a function ψ such that $\sup_x \mathfrak{d}_{\mathcal{Y}}\big(0, \psi(x)\big) \leq \bar{E}_h \in \mathbb{R}$.*

Assume we are given a trajectory $\Big(x_n\Big)_{n \in \mathbb{N}}$ of inputs that is bounded, i.e. where $\mathfrak{d}(x_n, 0) \leq \beta$ for some $\beta \in \mathbb{R}_+$ and all $n \in \mathbb{N}$. Furthermore, assume $\mathcal{D}_{n+1} = \mathcal{D}_n \cup \big\{\big(x_n, \tilde{f}(x_n)\big)\big\}$ and thus, $G_n = \{x_i | i \in \mathbb{N}, i < n\}$. Then the prediction error on the sequence vanishes up to the level of sample-consistency and Lipschitz continuity in the following sense:

$$\mathfrak{d}_{\mathcal{Y}}\big(\hat{\mathfrak{f}}_{\mathfrak{n}}(x_n), f(x_n)\big) \overset{n\to\infty}{\longrightarrow} [0, \frac{q}{2} + 2\bar{\mathfrak{e}} + 2\bar{E}_h].$$

In particular, in case the observations are error-free ($\tilde{f} = f$) and assuming the target is Lipschitz continuous then, when choosing $\lambda = 0$, the prediction error is guaranteed to vanish. That is,

$$\mathfrak{d}_{\mathcal{Y}}\big(\hat{\mathfrak{f}}_{\mathfrak{n}}(x_n), f(x_n)\big) \overset{n\to\infty}{\longrightarrow} 0.$$

Proof. Let $\xi_n \in \operatorname{argmin}_{g \in G_n} \mathfrak{d}(x_n, g)$ denote the nearest neighbour of x_n in $G_n = \{x_1, ..., x_{n-1}\}$.

Since sequence (x_n) is bounded, Lemma III.1 is applicable and hence: (i) $\lim_{n\to\infty} \mathfrak{d}(x_n, \xi_n) = 0$.

By Lemma II.4, we know that for all inputs $g \in G_n$ $\Big\|f(g) - \hat{\mathfrak{f}}_{\mathfrak{n}}(g)\Big\|_\infty \leq \frac{\lambda}{2} + \|\mathfrak{e}(g)\|_\infty \leq \frac{\lambda}{2} + \bar{\mathfrak{e}}$. Therefore, if we set $\lambda = 2\bar{\mathfrak{e}} + q$ then $\mathfrak{d}_{\mathcal{Y}}\big(\hat{\mathfrak{f}}_{\mathfrak{n}}(\xi_n), f(\xi_n)\big) \leq 2\bar{\mathfrak{e}} + \frac{q}{2}$. Hence, appealing to the triangle inequality, we see that (ii) $\mathfrak{d}_{\mathcal{Y}}\big(\hat{\mathfrak{f}}_{\mathfrak{n}}(x_n), f(\xi_n)\big) \leq \mathfrak{d}_{\mathcal{Y}}\big(\hat{\mathfrak{f}}_{\mathfrak{n}}(x_n), \hat{\mathfrak{f}}_{\mathfrak{n}}(\xi_n)\big) + 2\bar{\mathfrak{e}} + \frac{q}{2}$.

Moreover we note that the predictors $\hat{\mathfrak{f}}_{\mathfrak{n}}$ have Lipschitz constants $L(n)$ and that the $L(n)$ are bounded from above by some $\bar{L} \in \mathbb{R}$. Thus, (iii) $\exists \bar{L} \in \mathbb{R} \forall n \in \mathbb{N} : \hat{\mathfrak{f}}_{\mathfrak{n}} \in \mathrm{Lip}(\bar{L})$. In conclusion, $0 \leq \mathfrak{d}_{\mathcal{Y}}\big(\hat{\mathfrak{f}}_{\mathfrak{n}}(x_n), f(x_n)\big)$
$\leq \mathfrak{d}_{\mathcal{Y}}\big(\hat{\mathfrak{f}}_{\mathfrak{n}}(x_n), f(\xi_n)\big) + \mathfrak{d}_{\mathcal{Y}}\big(f(\xi_n), f(x_n)\big)$
$\overset{(ii)}{\leq} \mathfrak{d}_{\mathcal{Y}}\big(\hat{\mathfrak{f}}_{\mathfrak{n}}(x_n), \hat{\mathfrak{f}}_{\mathfrak{n}}(\xi_n)\big) + 2\bar{\mathfrak{e}} + \frac{q}{2} + \mathfrak{d}_{\mathcal{Y}}\big(f(\xi_n), f(x_n)\big)$
$\leq \mathfrak{d}_{\mathcal{Y}}\big(\hat{\mathfrak{f}}_{\mathfrak{n}}(x_n), \hat{\mathfrak{f}}_{\mathfrak{n}}(\xi_n)\big) + 2\bar{\mathfrak{e}} + \frac{q}{2} + \mathfrak{d}_{\mathcal{Y}}\big(\phi(\xi_n), \phi(x_n)\big) + 2\bar{E}_h$
$\overset{(iii)}{\leq} (\bar{L} + L^*)\,\mathfrak{d}(x_n, \xi_n) + 2\bar{\mathfrak{e}} + \frac{q}{2} + 2\bar{E}_h \overset{n\to\infty}{\longrightarrow} 2\bar{\mathfrak{e}} + \frac{q}{2} + 2\bar{E}_h$. □

IV. Application to Model-Reference Adaptive Control

A. Model reference adaptive control

Before proceeding with the application scenario, we will commence with (i) outlining model reference adaptive control (MRAC) [1] as considered in [9] and (ii) describe the deployment of LACKI to this framework.

(i) Assume $m \in \mathbb{N}$ to be the dimensionality of a configuration of the system in question and define $d = 2m$ to be the dimensionality of the pertaining state space $\mathcal{X}$.

Let $x = [x_1; x_2] \in \mathcal{X}$ denote the state of the plant to be controlled. Given the control-affine system

$$\dot{x}_1 = x_2, \;\dot{x}_2 = a(x) + b(x)\,u(x) \tag{5}$$

it is desired to find a control law $u(x)$ such that the closed-loop dynamics exhibit a desired reference behaviour: $\dot{\xi}_1 = \xi_2, \dot{\xi}_2 = f_r(\xi, r)$ where r is a reference command, f_r some desired response and $t \mapsto \xi(t)$ is the reference trajectory.

If a priori a and b are believed to coincide with $\hat{a}_0, \hat{b}_0$ respectively, the inversion control $u = \hat{b}_0^{-1}(-\hat{a}_0 + u')$ is applied. This reduces the closed-loop dynamics to $\dot{x}_1 = x_2, \dot{x}_2 = u' + \tilde{a}(x, u)$ where $\tilde{a}(x, u)$ captures the modelling error of the dynamics:

$$\tilde{a}(x, u) = a(x) - \hat{a}_0(x) + \big(b(x) - \hat{b}_0(x)\big)u. \tag{6}$$

Let $I_d \in \mathbb{R}^{d \times d}$ denote the identity matrix. If b is perfectly known, then $b - \hat{b}_0^{-1} = 0$ and the model error can be written as $\tilde{a}(x) = a(x) - \hat{a}_0(x)$. In particular, $\tilde{a}$ has lost its dependence on the control input.

In this situation [9], [8] propose to set the pseudo control as follows: $u'(x) := \nu_r + \nu_{pd} - \nu_{ad}$ where $\nu_r = f_r(\xi, r)$ is a feed-forward reference term, ν_{ad} is a yet to be defined output of a learning module *adaptive element* and $\nu_{pd} = [K_1 K_2]e$ is a feedback error term designed to decrease the *tracking error* $e(t) = \xi(t) - x(t)$ by defining $K_1, K_2 \in \mathbb{R}^{m \times m}$ as described in what is to follow.

Inserting these components, we see that the resulting *error dynamics* are:

$$\dot{e} = \dot{\xi} - [x_2; \nu_r + \nu_{pd} + \tilde{a}(x)] = Me + B\big(\nu_{ad}(x) - \tilde{a}(x)\big) \tag{7}$$

where $M = \begin{pmatrix} O_m & I_m \\ -K_1 & -K_2 \end{pmatrix}$ and $B = \begin{pmatrix} O_m \\ I_m \end{pmatrix}$. If the feedback gain matrices K_1, K_2 parametrising ν_{pd} are chosen such that M is stable then the error dynamics converge to zero as desired, provided the learning error E_λ vanishes: $E_\lambda(x(t)) = \|\nu_{ad}(x(t)) - a(x(t))\| \overset{t\to\infty}{\longrightarrow} 0$.

(ii) It is assumed that the adaptive element is the output of a learning algorithm that is tasked to learn $\tilde{a}$ online. This is done by continuously feeding it training examples of the form $\big(x(t_i), \tilde{a}(x(t_i)) + \varepsilon_i\big)$ where ε_i is observational noise.

Intuitively, assuming the learning algorithm is suitable to learn target $\tilde{a}$ (i.e. $\tilde{a}$ is close to some element in the hypothesis space [14] of the learner) and that the controller manages to keep the visited state space bounded, the learning error (as a function of time t) should vanish.

Substituting different learning algorithms yields different adaptive controllers. *RBFN-MRAC* [12] utilises radial basis function neural networks for this purpose whereas *GP-MRAC* employs Gaussian process learning [17] to learn $\tilde{a}$ [9], [8].

In what is to follow, we utilise our LACKI method as the adaptive element. Following the nomenclature of the previous methods we name the resulting adaptive controller *LACKI-MRAC*.

As mentioned above, the guarantee that the learning error vanishes over time can be translated into a guarantee of vanishing tracking error. For a discrete-time version of the MRAC setting, we can therefore appeal to Theorem III.2 to establish conditions under which LACKI-MRAC is guaranteed to eventually achieve tracking success. In particular, in the long version of this paper [4], we derive convergence guarantees, which, for the case of error-free observations, can be stated as follows:

Corollary IV.1 ([4])**.** *In the special case of error-free observations of a Lipschitz continuous target function and assuming bounded prediction errors, choosing a parameter $\lambda = 0$ implies that the tracking error vanishes, i.e. :*

$$\|e_n\|_\infty \overset{n\to\infty}{\longrightarrow} 0.$$

The control sequence $(u(x_n)_{n\in\mathbb{N}}$ converges if the reference trajectory $\left(\xi_n\right)_{n\in\mathbb{N}}$ does.

Note, the assumption of bounded prediction error can be achieved if either state space is bounded or, if we allow for unbounded control output which allows the linear control part to effectively bound the reachable set of states of the closed-loop dynamics.

B. Learning-based tracking control of an F-4 fighter jet under wing rock

As pointed out in [10], modern fighter aircraft designs are susceptible to lightly damped oscillations in roll known as "wing rock". Commonly occurring during landing [18], removing wing rock from the dynamics is crucial for precision control of such aircraft. Precision tracking control in the presence of wing rock is a nonlinear problem of practical importance and has served as a test bed for a number nonlinear adaptive control methods [9], [15], [10].

For comparison, we replicated the experiments of Chowdhary et. al. [9], [8].[1] Using a realistic model of the roll dynamics of an F-4 fighter jet, the authors examined the task of using a model-reference adaptive controller (MRAC) to perform a roll manoeuvre under uncertain wing rock. Within a time span between t_0 and t_f, the task was to control the aircraft's ailerons on order to cause the aircraft's state trajectory $x : [t_0, t_f] \to \mathbb{R}^2$ to closely follow a roll manoeuvre prescribed by the reference trajectory $\xi(\cdot)$. Here the first component of the state and reference was the roll angle and the second was the angular velocity.

Since wing rock can destabilise the dynamics, the authors proposed to utilise a Gaussian process approach to learn a model of the wing rock dynamics online and demonstrated this could significantly improve tracking performance over competing methods. They compared their Gaussian process based approach, called *GP-MRAC*, to the more established adaptive model-reference control approach based on RBF neural networks [19], [12], referred to as *RBFN-MRAC*. As the controller was meant to adapt to the uncertain wing rock dynamics online during runtime, computational real time constraints necessitated to fix the kernel hyper-parameters of the GP. Furthermore, they also proposed to limit the GP to a fixed budget of training examples which would be incrementally updated online.

Replacing the GP by our LACKI learner, we readily obtain an analogous learning-based controller which we call *LACKI-MRAC*. For baseline comparison, we also examined the performance of a simple PD-controller.

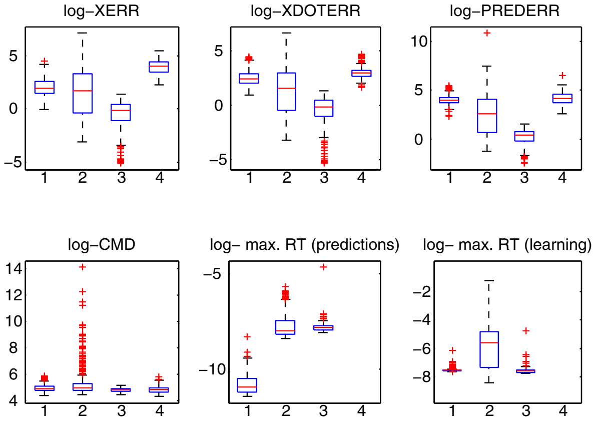

Fig. 2. Performance of the different online controllers over a range of 555 trials with randomised parameter settings and initial conditions. 1: RBFN-MRAC, 2: GP-MRAC, 3:LACKI-MRAC, 4: PD-Controller. LACKI-MRAC outperforms all other methods with respect to all performance measures, except for prediction run time (where the parametric learner RBFN-MRAC performs best).

We created 555 randomised test runs of the wing rock tracking problems and tested each control algorithm on each one of them. The initial state $x(t_0)$ was drawn uniformly at random from $[0, 7] \times [0, 7]$, the initial kernel length scales were drawn uniformly at random from $[0.05, 2]$, and used both for RBF-MRAC and GP-MRAC. For LACKI, we chose $\lambda = 0$. The parameter weights W of the system dynamics (cf. [9]) were multiplied by a constant drawn uniformly at random from the interval $[0, 2]$. To allow for better predictive performance of GP-MRAC, we set the maximal budget to 200 training examples (twice as large as in the experiments of [9]). The feedback gains of the linear pseudo controller were chosen to be $K_1 = K_2 = 1$ (see [9] for more explanations). As a baseline comparison, we also tested the performance of a simple $PD-$ controller with just these feedback gains.

The performance of all controllers across these randomised trials is depicted in Fig. 2. Each data point of each boxplot represent a performance measurement for one particular trial.

For each method, the figures show the boxplots of the following recorded quantities:

- *log-XERR*: cumulative angular position error (log-deg), i.e. $\log(\int_{t_0}^{t_f} \|\xi_1(t) - x_1(t)\| \, dt\,)$.
- *log-XDOTERR*: cumulative roll rate error (log-deg/sec.), i.e. $\log(\int_{t_0}^{t_f} \|\xi_2(t) - x_2(t)\| \, dt\,)$.
- *log-PREDERR*: log-prediction error, i.e. $\log(\int_{t_0}^{t_f} \left\| \hat{\mathfrak{f}}_n(x(t)) - f(x(t)) \right\| \, dt\,)$ where f is a vector field affected by the wing rock.
- *log-CMD*: cumulative control magnitude (log-scale), i.e. $\log(\int_{t_0}^{t_f} \|u(t)\| \, dt\,)$.
- *log-max. RT (predictions)*: the log of the maximal run time (within time span $[t_0, t_f]$) each method took to generate a prediction ν_{ad} within the time span.

[1] We are grateful to the authors for kindly providing their code.

- *log-max. RT (learning)*: the log of the maximal run time (within time span $[t_0, t_f]$) it took each method to incorporate a new training example of the drift $\tilde{a}$.

Discussion: All three adaptive methods outperformed the simple $PD-$ controller in terms of tracking error. With regard to prediction run time, RBFN-MRAC outperformed both GP-MRAC and LACKI-MRAC. This is hardly surprising. After all, RBFN-MRAC is a parametric method with constant prediction time. By contrast, both non-parametric methods will have prediction times growing with the number of training examples. That is, it would be the case if GP-MRAC were given an infinite training size budget. Indeed one might argue whether GP-MRAC, if operated with a finite budget, actually is a parametric approximation where the parameter consists of the hyper-parameters along with the fixed-size training data matrix. When comparing the (maximum) prediction and learning run times one should also bear in mind that GP-MRAC predicted with up to 200 examples in the training data set. By contrast, fast enough to process large online data, LACKI-MRAC undiscerningly had incorporated all 10001 training points by the end of each trial. Across the remaining metrics, LACKI-MRAC markedly outperformed all other methods.

V. Conclusions

We have introduced *Lazily Adapted Constant Kinky Inference (LACKI)* as an approach to nonparametric machine learning. Our method was built on the framework of *Kinky Inference* which is a generalisation of well-known approaches such as LI and NSM methods that have become popular in numerical mathematics and learning-based control. Our approach inherits the numerical simplicity of these methods but does not require a priori knowledge of a Lipschitz constant of the underlying target function. Of course, this is of great practical interest since it endows LACKI with substantially improved black-box learning capabilities. In contrast to competing NSM approaches based on Lipschitz constant estimation [13], [5], LACKI is fast enough to support online learning and, we can still give theoretical guarantees on the learning performance. Being a nonparametric regression method that is simple but can learn rich function classes, we believe LACKI hits a sweet spot between robustness and efficiency on the one hand and high learning capacity on the other. Furthermore, even with the hyper-parameter estimator in place, it is fast enough to be utilised in an online learning setting. This is in contrast to other methods, for instance in Gaussian process regression, that rely on hyper-parameter optimisation but which are burdened with more extensive computational effort. These computational advantages, allows LACKI to be utilised in online learning and model-reference adaptive control. For these scenarios we were able to give learning guarantees which can be converted into guarantees on tracking success.

Our theoretical guarantees assume the observational errors to be bounded by some $\bar{\mathfrak{e}} \in \mathbb{R}_{\geq 0}$ and that the hyper-parameter λ is set to at least two times $\bar{\mathfrak{e}}$. Knowledge of such a bound is a common assumption in learning-based control [7], [2]. And, other common assumptions, such as white-noise disturbances, are physically unrealistic. Nonetheless, in practice, such a noise bound may be unknown or be too large to give good convergence behavior of our method. Ongoing work investigates how to estimate the $\bar{\mathfrak{e}}$ parameter from the data: In the presence of bounded i.i.d., additive, stochastic noise this could be done by using POKI-LC [5] on a small patch of the input space to obtain a local estimate of the Lipschitz constant. The noise bound would then be computed as a function of the worst-case error of the POKI-LC predictor on a test sample.

References

[1] K. J. Aström and B. Wittenmark. *Adaptive Control*. Addison-Wesley, 2nd edition, 2013.

[2] A. Aswani, H. Gonzalez, S. S. Sastry, and C. Tomlin. Provably safe and robust learning-based model predictive control. *Automatica*, 2013.

[3] G. Beliakov. Interpolation of Lipschitz functions. *Journal of Computational and Applied Mathematics*, 2006.

[4] J. Calliess. Lazily Adapted Constant Kinky Inference for Nonparametric Regression and Model-Reference Adaptive Control. *Arxiv preprint arXiv:1701.00178*, 2016.

[5] J. Calliess. Lipschitz Optimisation for Lipschitz Interpolation. In *American Control Conference*, 2017.

[6] Jan-Peter Calliess. *Conservative decision-making and inference in uncertain dynamical systems*. PhD thesis, University of Oxford, 2014.

[7] M. Canale, L. Fagiano, and M. C. Signorile. Nonlinear model predictive control from data: a set membership approach. *Int. J. Robust Nonlinear Control*, 2014.

[8] G. Cho, G. Chowdhary, A. Kingravi, J. P. . How, and A. Vela. A Bayesian nonparametric approach to adaptive control using Gaussian processes. In *CDC*, 2013.

[9] Girish Chowdhary, H.A. Kingravi, J.P. How, and P.A. Vela. Bayesian nonparametric adaptive control using Gaussian processes. Technical report, MIT, 2013.

[10] Girish Chowdhary, Hassan A. Kingravi, Jonathan How, and Patricio A. Vela. Nonparametric adaptive control of time-varying systems using Gaussian processes. In *American Control Conference (ACC)*, 2013.

[11] W. S. Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association*, 1979.

[12] Y. H. Kim and F. Lewis. High-level feedback control with neural networks. *Robotics and Intelligent Systems*, 1998.

[13] M. Milanese and C. Novara. Set membership identification of nonlinear systems. *Automatica*, 2004.

[14] T. Mitchell. *Machine Learning*. Mc Graw Hill, 1997.

[15] M.M. Monahemi and M. Krstic. Control of wingrock motion using adaptive feedback linearization. *J. of. Guidance Control and Dynamics.*, 1996.

[16] E. A. Nadaraya. On estimating regression. *Theory of Probability and its Applications.*, 1964.

[17] C.E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[18] A. A. Saad. *Simulation and Analysis of wing rock physics for a generic fighter model with three degrees of freedom.* PhD thesis, Air Force Institute of Technology, Air University, 2000.

[19] R. Sanner and J.-J. Slotine. Gaussian networks for direct adaptive control. *Trans. on Neural Networks*, 1992.

[20] R. G. Strongin. On the convergence of an algorithm for finding a global extremum. *Engineering in Cybernetics*, 1973.

[21] A.G. Sukharev. Optimal method of constructing best uniform approximation for functions of a certain class. *Comput. Math. and Math. Phys.*, 1978.

[22] G. S. Watson. Smooth regression analysis. *Sankhya: The Indian Journal of Statistics*, 1964.

[23] Z. B. Zabinsky, R. L. Smith, and B. P. Kristinsdottir. Optimal estimation of univariate black-box Lipschitz functions with upper and lower bounds. *Computers and Operations Research*, 2003.