Localised Kinky Inference

A. Blaas[1], J.M. Manzano[2], D. Limon[2] and J. Calliess[1]

Abstract— **Their flexibility to learn general function classes renders nonparametric regression algorithms particularly attractive in system identification and data-based control settings, where little a priori knowledge about a dynamical system is to be presumed. Building on approaches known as NSM- or Lipschitz regression, we propose a new nonparametric machine learning approach. While it inherits theoretical learning guarantees from the methods it is built upon, it is designed to limit the computational effort both for learning and for generating predictions. This renders our method applicable to online system identification and control settings where the desired sample frequency precludes previous nonparametric approaches from being deployed. Apart from deriving a guarantee on the ability of our method to learn any continuous function, we illustrate some of its practical merits on a number of benchmark comparison problems.**

I. INTRODUCTION

In system identification and data-based control, machine learning algorithms have gained increasing popularity. They promise increased flexibility to learn and predict in rich classes of dynamical systems. If learning and prediction can be performed with sufficient frequency, this would allow a controller to achieve its objective even if the plant dynamics change or are static, but substantially deviate from an a priori model. Among the manifold learning methods available, nonparametric regression approaches offer the greatest flexibility and typically are guaranteed to be able to learn the widest classes of functions in the infinite data limit [13], [25].

Unfortunately, their computational complexity for prediction tends to increase with the observed data. Furthermore, most practically viable techniques for nonparametric regression, such as the Nadaraya-Watson estimator [19], [26], the LOESS method [10], Lipschitz interpolation [23] or Gaussian processes (GPs) [20], suffer from a practical limitation: their regression performance critically depends on an a priori choice of hyper-parameters.

To make these approaches perform well in practice, training methods typically involve hyperparameter tuning which is often performed either manually or by employing optimisers which then effectively become part of the learning algorithm.

In control-related reinforcement learning applications, such approaches can often yield impressive practical performance [11], [12]. Unfortunately though, the optimisation problems typically are highly non-convex. This means that the quality of the data-dependent hyperparameter and hence, of the learning outcomes, are typically poorly understood. By extension, this renders the problem of properly analysing the performance of the data-based controller an elusive one.

Furthermore, to get sensible outcomes, the optimisers will have to be allowed to run for a long time each time they are presented with new data. This substantially exacerbates the computational training effort, further denying the application of such methods to system identification and online control in all but very slowly evolving dynamical systems.

Recent developments that have sought to address these limitations were built on a class of nonparametric learning methods that have been referred to as *Kinky Inference* [5]. This class, which encompasses Lipschitz interpolation [23] and NSM-methods [16] allows for an easier theoretical analysis in control [5], [6]. Their computations are simple enough to be run on an embedded system, and their hyperparameter estimation is computationally tractable enough to allow for theoretical guarantees on the outcomes, which in turn can be extended to guarantees of the data-based controllers [3], [15], [14].

In the context of data-based model-predictive control, [15] designed a scheme to reduce prediction effort by partitioning the data space into hyperrectangles. A prediction of a query input would then be based on training examples contained in the same or adjacent hyperrectangles. By managing the size of the rectangles appropriately, the authors could provide a probabilistic bound on the computational prediction error under distributional assumptions on the data. They have also demonstrated how this approach can render the approach tractable in MPC not only in theory, but also in practice. However, their work assumed offline learning only. That is, the learner was trained on a fixed sample of data collected before the controller was applied. This was necessary, since their hyperparameter estimator was based on a modified Strongin estimate of a global Lipschitz constant which allowed them to link to existing theoretical convergence analysis of the resulting regressor as presented in [3]. And unfortunately, the computational effort for estimating the global Lipschitz constant hyperparameter still grows with the number of data points.

In this work, we will address this deficiency. Instead of computing one global Lipschitz hyperparameter, we will maintain a set of local Lipschitz constant hyperparameter estimates predictions will be based on. This promises to have two benefits: Firstly, it allows us to bound computational effort for training (provided we can bound the number of training examples to be considered for the computation of each constant and project onto this set in constant time). Secondly, it can improve prediction performance on locally Lipschitz target functions that have local Lipschitz constants

[1]Dept. of Engineering Science, Oxford University, UK
[2]Dept. of Systems Engineering and Automation, Seville University, Spain

that vary substantially in different parts of input space. Since our method is a kinky inference regression approach that relies on fully localised information only, we will refer to it has *Localised Kinky Inference (LoKI)*.

Normally, nonparametric learning methods offer extensive theoretical guarantees to learn continuous functions without little a priori knowledge, whereas parametric approaches are more representationally restricted but tend to be fast enough to be deployed in online learning and control. In the remainder of this paper, we introduce our LoKI method and show that it can offer the best of both worlds: On the one hand, we show how to provide learning-theoretic worst-case guarantees, allowing us to be confident we can learn any continuous function. On the other, we show how our method can be set up to have bounded computational complexity, both for training and prediction, rendering our approach amenable to online learning-based control settings with higher sampling frequency than could have been achieved with previous nonparametric learning methods. We compare the performance of our regression method to competing learning methods on a number of standard benchmark data sets. Moreover, we illustrate the merits of our approach in a simple online learning-based flight manoeuvre control task previously considered as a test bed in the literature [7], [3], [4].

II. Kinky Inference and Lipschitz Interpolation

The term *kinky inference (KI)* was introduced in [5] to describe a class of nonparametric regression rules which we will briefly rehearse in this section. The regression task under consideration is to learn a continuous *target function* $f : \mathcal{W} \to \mathcal{Z}$.

It is assumed that f is Hölder continuous with constant L^* and exponent $0 < \alpha \leq 1$ relative to the input and output space pseudo-metrics $\mathfrak{d}$ and $\mathfrak{d}_{\mathcal{Z}}$. That is, for all $w_1, w_2 \in \mathcal{W}$ we have $\mathfrak{d}_{\mathcal{Z}}(f(w_1) - f(w_2)) \leq L^* \, \mathfrak{d}(w_1, w_2)^{\alpha}$. For the special case $\alpha = 1$, f is called Lipschitz continuous and while our theoretical analysis readily extends to the general Hölder case, we shall assume $\alpha = 1$ for the rameinder of this work (note, any continuous function is Lipschitz up to an arbitrarily small error, so that our framework addresses learning of any continuous function). Also, for ease of notation, but without loss of generality, we will restrict our exposition in the remainder of this work to the case where f is real-valued, i.e. $\mathcal{Z} = \mathbb{R}$ and $\mathfrak{d}_{\mathcal{Z}}(y, y') = |y - y'|$. Regression is the task of predicting the target function at unobserved query inputs on the basis of a (possibly noisy) sample or data set. So, assume we have access to a sample or data set

$$\mathcal{D}_n \ : \ = \ \{(w_i, \tilde{f}(w_i)) | i = 1, \ldots, n\}$$

where $\tilde{f}$ denotes a noisy version of f (note, we will use $\tilde{f}_i$ and $\tilde{f}(w_i)$ interchangeably). The set containing only the input data points is denoted as $\mathcal{W}_{\mathcal{D}_n} = \mathrm{Proj}_{\mathcal{W}}(\mathcal{D}_n)$. For our theoretical analysis (but not necessarily our practical applications), we assume the noise to be bounded by some $\bar{\mathfrak{e}} \geq 0$. That is, we will suppose $\forall w \in \mathcal{W} : \ \mathfrak{d}(\tilde{f}(w), f(w)) \leq \bar{\mathfrak{e}}$. The regression task then is to utilise $\mathcal{D}_n$ to compute a prediction $\hat{\mathfrak{f}}_n(q)$ of $f(q)$ for any *query* input $q \in \mathcal{W}$.

While there are a great many methods to compute such a prediction we will build upon the following class of regressors:

Definition 1 (Kinky inference (KI) rule [5] - simplified): Let $q \in \mathcal{W}$ and $\mathcal{D}_n$ defined as before . We define

$$\mathfrak{u}_n\big(q; \theta(n)\big) \ := \ \min_{w_i \in \mathcal{W}_{\mathcal{D}_n}} \tilde{f}_i + \mathfrak{d}(q, w_i; \theta(n)), \qquad (1)$$

$$\mathfrak{l}_n\big(q; \theta(n)\big) \ := \ \max_{w_i \in \mathcal{W}_{\mathcal{D}_n}} \tilde{f}_i - \mathfrak{d}(q, w_i; \theta(n)). \qquad (2)$$

as upper and lower bound functions from which we construct the kinky inference predictor $\hat{\mathfrak{f}}_n\big(\cdot; \theta(n), \mathcal{D}_n\big) : \mathcal{W} \to \mathcal{Z}$ to perform inference over function values at query inputs q as per:

$$\hat{\mathfrak{f}}_n\big(q; \theta(n), \mathcal{D}_n\big) := \frac{1}{2}\mathfrak{u}_n\big(q; \theta(n)\big) + \frac{1}{2}\mathfrak{l}_n\big(q; \theta(n)\big). \qquad (3)$$

Note, the parameter $\theta(n)$ of the pseudo-metric is a hyperparameter of the regression method. In the case where $\mathfrak{d}(x, y; \theta(n)) = \ell(n) \, \|x - y\|$, which is often referred to as *Lipschitz Interpolation* [2] or as *Nonlinear Set Interpolation* [16], this (hyper-)parameter $\theta(n) = \ell(n)$ is the supposed Lipschitz constant of the target. If L^* is known, it can be used as $\ell(n)$. However when it is unknown, $\ell(n))$ it has to be learned from the data, e.g. lazily adapted [22] as $\ell(n) := \max_{i \neq j} \frac{|\tilde{f}_i - \tilde{f}_j|}{\|w_i - w_j\|}$. Unfortunately, this estimate has the problem of being unbounded in the presence of observational noise. In the case of bounded noise, [5], [3], [4] proposed to use the alternative estimator

$$\ell(n) := \max_{w_i, w_j \in \mathcal{W}_{\mathcal{D}_n} | i \neq j} \frac{\left|\tilde{f}_i - \tilde{f}_j\right| - 2\bar{\mathfrak{e}}}{\|w_i - w_j\|} \qquad (4)$$

where $\bar{\mathfrak{e}} \in \mathbb{R}$ is an upper bound on the (zero-mean) level of noise. This KI method, referred to as LACKI, has been shown to converge to any continuous target in the infinite data limit up to an arbitrarily adjustable low worst-case error [3] and it has shown good performance in a number of control problems [5], [4]. However, it has two major drawbacks. Firstly, the time it takes to calculate $\ell(n)$ is quadratic in the number of sample points n, which for large data sets is prohibitive. Similarly, the prediction cost being linear in the number of sample points can be prohibitive especially in control settings which require frequent predictions during planning updates. Secondly, the resulting estimator does not take into account local differences in smoothness of the target function as $\ell(n)$ is estimated globally.

III. Localised Kinky Inference

Going back to a more general, non-simplified definition of the KI rule [5], we can define a version of the KI predictor which overcomes the two drawbacks of LACKI and other simplified KI approaches described above. The key element is based in the idea of only cosidering a certain subset $\mathcal{I}_n(q) \subset \{1, \ldots n\}$ of indices to construct the predictor in Eq. (3), as has been proposed in [5], [4]. Our main contribution is

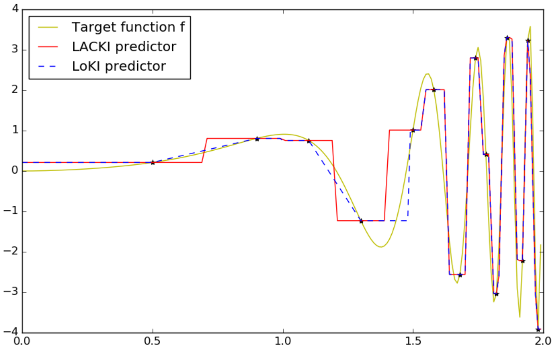

Fig. 1. Illustration of the advantage of learning the Lipschitz constant locally. The target function $f(x) = x^2 \sin(2x^{2x})$ (yellow) as predicted by LACKI (orange) and LoKI with 3 sub-hypercubes (dotted blue). While both predictors coincide in the rightmost sub-hypercube, they diverge in the left sub-hypercubes as LACKI overestimates the (local) Lipschitz constant and predicts further off the true target function whose local smoothness is better estimated by LoKI.

to extend this idea and also use an estimator for the Lipschitz constant which is not estimated globally as in Eq. (4), but only from points in $\mathcal{I}_n(q)$. Defining $\mathcal{I}_n(q)$ to only contain those indices whose sample points are in a neighbourhood of q, we call the resulting KI method Localised Kinky Inference (LoKI).

Formally, the LoKI predictor is defined as

$$\begin{aligned}\hat{\mathfrak{f}}(q;\theta(q),\mathcal{D}_n) &= \frac{1}{2}\min_{i\in\mathcal{I}_n(q)}\left(\tilde{f}_i + L_{\mathcal{D}_n}(q)\|q-w_i\|\right)\\ &+\frac{1}{2}\max_{i\in\mathcal{I}_n(q)}\left(\tilde{f}_i - L_{\mathcal{D}_n}(q)\|q-w_i\|\right), \quad (5)\end{aligned}$$

where

$$L_{\mathcal{D}_n}(q) := \max_{(w_i,w_j)|i,j\in\mathcal{I}_n(q),i\neq j}\frac{\|\tilde{f}(w_i)-\tilde{f}(w_j)\|-\gamma}{\|w_i-w_j\|} \quad (6)$$

with $\gamma = 2\bar{\mathfrak{e}}$. We write $\hat{\mathfrak{f}}_n(q)$ shorthand for $\hat{\mathfrak{f}}(q;\theta(q),\mathcal{D}_n)$. While LACKI uses all available data points to learn the hyperparameter and to predict at a new query point, our method only uses the subset $\mathcal{I}_n(q)$ to do so, which significantly reduces the computation time while simultaneously preventing the model from being overly conservative in particularly smooth regions. Fig. 1 illustrates this in comparison to the global (LACKI) estimation.

A. Defining Hyperrectangles

So far, we have not specified $\mathcal{I}_n(q)$. In order to do so, we partition the input space $\mathcal{W} \subset \mathbb{R}^d$ into several $\mathcal{W}_k$, such that the union of them conform the original space $\mathcal{W} = \cup(\mathcal{W}_k)$, and their intersection is null: $\cap_{k\neq j}\mathcal{W}_k = \emptyset$, $\forall j$.

Accordingly, $\mathcal{D}_n$ is partitioned into the same amount of nonintersecting subsets of sample points. This partition is calculated offline and then a classification algorithm is built to locate the partition $\mathcal{W}_k$ to which a query point q belongs to. $\mathcal{I}_n(q)$ is then composed of the indices of all data points which lie in $\mathcal{W}_k$.

The partition can be calculated taking into account different objectives, as for instance, to ensure a regular distribution of data points. For the remainder of this work, we will define the set of the partitions to be equally sized hyperrectangles. Assuming that all data lies in some original hyperrectangle in input space of the form $H = \{w : w_{(l)} \in [\mathfrak{a}_{(l)}, \mathfrak{b}_{(l)}], \forall l = 1, \ldots, d\}$ (once the samples are drawn, we can always ensure this by appropriate rescaling of the workspace), we partition H into m^d (sub-)hyperrectangles H_k indexed by the tuple $k := (k_1, ..., k_d)$. where $k_l \in \{1, ..., m\}$ for $(l = 1, ..., d)$ and define

$$\begin{aligned}H_k = \quad &\Bigg\{w : k_l = \min\{m, \max\{1, \left\lfloor\frac{m(w_{(l)}-\mathfrak{a}_{(l)})}{\mathfrak{b}_{(l)}-\mathfrak{a}_{(l)}}\right\rfloor + 1\}\},\\ &\forall l = 1, ..., d\Bigg\}.\end{aligned}$$

These (sub-)hyperrectangles then define our partition, i.e.

$$\mathcal{W}_k = H_k \quad (7)$$

B. Computational Complexity

We will now evaluate the computational complexity of the proposed model under the assumption that data points are i.i.d. uniformly drawn in H. This will be done separately for training and prediction.

1) *Training*

For any given point $w \in H_k$, computing the index $k = (k_1, ..., k_d)$ can be done in $\mathcal{O}(d)$ basic computational steps. This means that before any Lipschitz constant can be learned locally, $\mathcal{O}(nd)$ computations are necessary to assign all training points to their corresponding hyperrectangles. Subsequently, learning the local Lipschitz constant L_k of H_k requires $\mathcal{O}({n_k}^2)$ computational steps, where n_k is the number of samples contained in H_k. So, training can be done in

$$T^{train}_{n,m,d} := c_1\sum_{k=1}^{m^d}(c_2+n_k^2)1_{\{\exists w_i\in\mathcal{W}_k\}} + c_3 n\, d + c_4$$

computational steps for some (algorithm-dependent) parameters c_1, c_2, c_3, c_4. Given the uniform i.i.d. distribution of the sample points in $\mathcal{W}$, n_k is a random variable following a binomial distribution with success probability $p = m^{-d}$
(the probability for any single sample point to be in H_k) and number of trials n. Hence,

$$\mathbb{E}[n_k^2] = \frac{n^2}{m^{2d}} + \frac{n}{m^d} + \frac{n}{m^{2d}}.$$

Therefore, for $\Xi := \mathbb{E}[T^{train}_{n,m,d}]$

$$\begin{aligned}\Xi \quad &= \quad \mathbb{E}\left[c_1\sum_{k=1}^{m^d}(c_2+n_k^2)1_{\{\exists w_i\in\mathcal{W}_k\}}\right]\\ &\qquad\qquad + c_3 n\, d + c_4\\ &\leq \quad c_1\left(nc_2 + \frac{n^2+n}{m^d} + n\right) + c_3 n\, d + c_4\\ &\leq \quad c_1\left(n(c_2+2)+1\right) + c_3 n\, d + c_4,\end{aligned}$$

for large enough m (such that $m^d \geq n$). Thus, $\mathbb{E}[T^{train}_{n,m,d}] \in \mathcal{O}(nd)$.

2) *Prediction*
The number $T^{predict}_{n,m,d}$ of computational steps required for prediction is the same as for the projected KI method introduced in [4]. We have

$$\mathbb{E}[T^{predict}_{n,m,d}] = c_5 m^{-d} n + c_6 d + c_7,$$

which makes $E[T^{predict}_{n,m,d}] \in \mathcal{O}(m^{-d}n + d)$

These results can be utilised to answer the question of how to choose parameters m to control the average and maximal computational effort per query point.

C. *Consistency Guarantees*

Despite the reduced computational complexity compared to global KI approaches, we will now show that the LoKI predictor shares important desirable properties of the global approaches, e.g. sample consistency, or vanishing worst-case prediction error in the limit of infinitely dense data. While it turns out that the prediction model no longer is globally (Lipschitz) continuous, it will be shown that (Lipschitz) continuity still holds almost everywhere and that any discontinuity can be bounded. Some of the proofs are straight-forward extensions of the ones given for the LACKI method [3]. In the interest of brevity, we restrict ourselves proof sketches in those cases.

Lemma 1 (Sample-consistency): For any $w_i \in \mathcal{W}_{\mathcal{D}_n}$, we have $\|\hat{\mathfrak{f}}_n(w_i) - f(w_i)\| \leq \frac{\gamma}{2} + \bar{\mathfrak{e}}$

Proof. The proof is completely analogous to that of Lemma 2.7 in [3], where one only needs to replace $\mathcal{D}_n$ with $\mathcal{D}^{w_i}_n := \{(w_j, z_j) \in \mathcal{D}_n | j \in \mathcal{I}_n(w_i)\}$ to apply to our case.

Lemma 2 (Worst case error of LoKI): Under the assumption that f is Hölder continuous with constant L^*, the following guarantee holds for the worst case prediction error of LoKI with partition defined as in (7): For any $q \in \mathcal{W}$, we have

$$\begin{aligned} \|\hat{\mathfrak{f}}_n(q) - f(q)\| &\leq (L_{\mathcal{D}_n}(q) + L^*)\|q - \xi^q_n\| && (8)\\ &\quad + \frac{\gamma}{2} + \bar{\mathfrak{e}} && (9)\\ &\leq 2L^*\|q - \xi^q_n\| + \frac{\gamma}{2} + \bar{\mathfrak{e}} && (10)\end{aligned}$$

with $\xi^q_n \in \arg\min_{w_i \in H_k(q)} \|w_i - q\|$ being the nearest neighbour of q in the same hypercube (among the sample data points).

Proof. Again, the proof is very similar to the equivalent statement for the LACKI method provided in [3]. Let $\Delta := \|\hat{\mathfrak{f}}_n(q) - f(q)\|$. Applying the triangle inequality entails:

$$\begin{aligned} \Delta &\leq \|\hat{\mathfrak{f}}_n(q) - \hat{\mathfrak{f}}_n(\xi^q_n)\| + \|\hat{\mathfrak{f}}_n(\xi^q_n) - f(\xi^q_n)\|\\ &\quad + \|f(\xi^q_n) - f(q)\|\\ &\overset{(i)}{\leq} \|\hat{\mathfrak{f}}_n(q) - \hat{\mathfrak{f}}_n(\xi^q_n)\| + \|\hat{\mathfrak{f}}_n(\xi^q_n) - f(\xi^q_n)\|\\ &\quad + L^*\|\xi^q_n - q\|\\ &\overset{(ii)}{\leq} \|\hat{\mathfrak{f}}_n(q) - \hat{\mathfrak{f}}_n(\xi^q_n)\| + \frac{\gamma}{2} + \bar{\mathfrak{e}}\\ &\quad + L^*\|\xi^q_n - q\|\\ &\overset{(iii)}{\leq} L_{\mathcal{D}_n}(q)\|q - \xi^q_n\| + \frac{\gamma}{2} + \bar{\mathfrak{e}}\\ &\quad + L^*\|\xi^q_n - q\|\\ &\overset{(iv)}{\leq} 2L^*\|\xi^q_n - q\| + \frac{\gamma}{2} + \bar{\mathfrak{e}}\end{aligned}$$

Here, (i) follows from the Lipschitz continuity of f, (ii) follows from Lemma 1 and (iii) follows from Lemma 2.6 in [3] because by definition of ξ^q_n we have $\mathcal{I}_n(q) = \mathcal{I}_n(\xi^q_n)$. Finally it is easy to see from Eq. 6 that $L_{\mathcal{D}_n}(q) \leq \max_{(w_i,w_j)|i,j\leq n, i\neq j} \frac{\|\tilde{f}(w_i) - \tilde{f}(w_j)\| - \gamma}{\|w_i - w_j\|}$. This expression has been shown in Remark 2.2 in [3] to be smaller or equal to L^*, from which (iv) follows and the claim is proven.

On the basis of Lemma 2, it is straight-forward to establish that our method is capable of learning any Lipschitz continuous function in the limit of increasing data density:

Corollary 1 (Consistency of the LoKI predictor): If the grid of sample points becomes pointwise dense in $\mathcal{W}$ as $n \to \infty$, the pointwise prediction error vanishes up to $\frac{\gamma}{2} + \bar{\mathfrak{e}}$, i.e. $\|\hat{\mathfrak{f}}_n(q) - f(q)\| \to [\frac{\gamma}{2} + \bar{\mathfrak{e}}]$ for any $q \in \mathcal{W}$.

Lemma 3 (Boundedness of discontinuities): Given $\mathcal{D}_n$ and the partition defined in (7), any discontinuity of the predictor is bounded by $c = 4L^* z(m, \|\cdot\|) + \gamma + 2\bar{\mathfrak{e}} < \infty$, where z is a bounded function that maps the number of hyperrectangles per dimension, m, and the norm used, $\|\cdot\|$, onto the maximum possible metric distance between two points in the same hyperrectangle under that norm (and is therefore decreasing in m).

Proof (sketch). For any $q_1, q_2 \in \mathcal{W}$ we have for $\Lambda = \|\hat{\mathfrak{f}}_n(q_1) - \hat{\mathfrak{f}}_n(q_2)\|$ (using triangle inequality and Lemma 2)

$$\begin{aligned} \Lambda &\leq \|f(q_1) - \hat{\mathfrak{f}}_n(q_2)\| + \|\hat{\mathfrak{f}}_n(q_1) - f(q_1)\|\\ &\leq \|f(q_1) - f(q_2)\| + \|f(q_2) - \hat{\mathfrak{f}}_n(q_2)\|\\ &\quad + 2L^*\|q_1 - \xi^{q_1}_n\|^\alpha + \frac{\gamma}{2} + \bar{\mathfrak{e}}\\ &\leq 2L^*(\|q_1 - \xi^{q_1}_n\|^\alpha + \|q_2 - \xi^{q_2}_n\|^\alpha)\\ &\quad + \gamma + 2\bar{\mathfrak{e}} + \|f(q_1) - f(q_2)\|\\ &\leq 4L^* z(m, \|\cdot\|) + \gamma + 2\bar{\mathfrak{e}}\\ &\quad + \|f(q_1) - f(q_2)\|\end{aligned}$$

By the Lipschitz continuity of f, the last term $\|f(q_1) - f(q_2)\|$ vanishes as $\|q_1 - q_2\| \to 0$ and the claim follows.

Lemma 4 (Occurrence of discontinuities): Given $\mathcal{D}_n$ and the partition defined in (7), the Lebesgue measure of the set of possible discontinuities in the predictor $\hat{\mathfrak{f}}_n$ is 0, i.e. for $S_n := \{q \in \mathcal{W} | \exists \epsilon > 0 \forall \delta > 0 \exists q' \in \mathcal{W} : \|q - q'\| < \delta, \|\hat{\mathfrak{f}}_n(q) - \hat{\mathfrak{f}}_n(q')\| \geq \epsilon\}$ we have $\lambda(S_n) = 0$.

Proof. Inside any hyperrectangle H_k, the LoKI predictor corresponds to the global KI predictor which is known to be Lipschitz continuous and thereby continuous [23]. Discontinuities can therefore only appear on the borders of the hyperrectangles, which constitute $d-1$-dimensional hyperplanes. It is a standard result in measure theory that $d-1$-dimensional hyperplanes have 0 $d-$dimensional Lebesgue measure.

From this last result it follows that under any any probability measure built on top of the Lebesgue measure, there is zero probability of sampling from the boundaries between the partitioning hyperrectangles, and therefore any query point is surrounded with probability 1 by an epsilon ball where $\hat{f}_n$ is continuous.

While these last results show that the discontinuities that can occur in theory using LoKI are not to be expected to cause problems in practice, it is possible to eliminate them altogether by also considering the adjacent hyperrectangles for both estimation of the local Lipschitz constant and prediction. To such an end, $\mathcal{W}_k$ would be defined as

$$\mathcal{W}_k := \{H_{k+\beta} | \beta \in \{-1, 0, 1\}^d\}, \tag{11}$$

which can be shown to imply continuity of the resulting LoKI predictor $\hat{f}_n$. However, as a trade-off, this increases computational complexity by a factor of $\mathcal{O}(3^d)$ and thereby makes the method less suited for high-dimensional problems, which is why in the following experimental section we adhere to the original definition of $\mathcal{W}_k$ in (7).

IV. Experiments

In this section, we compare our approach to a number of well-established machine learning methods on problems in both control (*A.*) and standard pattern recognition (*B.*).

A. Model-Reference Adaptive Flight Control under Wingrock

We conceived our LoKI approach, bearing in mind the requirements in adaptive control settings. Here, we want to deploy learning methods that can learn and predict online with sufficient rapidity, to inform control actions with the desired sample frequency. To test our method against these requirements in comparison to alternative algorithms, we will apply it to a simple benchmark tracking control problem that has previously been considered in the literature ([4], [7], [9]) as a test bed for learning-based model reference adaptive controllers.

We will commence with 1) briefly reviewing model reference adaptive control (MRAC) [1] as considered in [8] and explaining the deployment of kinky inference to this framework before proceeding with 2) the description of the test bed and our benchmark results.

1) Model reference adaptive control and KI: We will now rehearse the description of MRAC for second-order systems following [8] and their notation as long as it does not cause confusion with earlier defined variables.

Assume $s \in \mathbb{N}$ to be the dimensionality of a configuration of the system in question and define $h = 2s$ 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

$$\dot{x}_1 = x_2, \ \dot{x}_2 = a(x) + b(x)\, u(x) \tag{12}$$

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:

$$\tilde{a}(x, u) = a(x) - \hat{a}_0(x) + \big(b(x) - \hat{b}_0(x)\big)u. \tag{13}$$

Let $I_h \in \mathbb{R}^{h \times h}$ 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 [8], [7] 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:

$$\dot{e} = \dot{\xi} - [x_2; \nu_r + \nu_{pd} + \tilde{a}(x)] = Me + B\big(\nu_{ad}(x) - \tilde{a}(x)\big) \tag{14}$$

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 parameterising 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, i.e. if $E_\lambda(x(t)) = \|\nu_{ad}(x(t)) - a(x(t))\| \overset{t\to\infty}{\longrightarrow} 0$.

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 $\big(x(t_i), \tilde{a}(x(t_i)) + \varepsilon_i\big)$. Here ε_i is observational noise, we assume the states $x(t_i)$ are observable and serve as inputs to our learned prediction model about a (thus, to link to our previous notation $\mathcal{X} = \mathcal{W}, x_i = w_i$, $a = f$ and $\tilde{a}(x(t_i)) + \varepsilon_i = \tilde{f}_i$).

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 [17] 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, e.g. *GP-MRAC* employs Gaussian process learning [20] to learn $\tilde{a}$ ([8], [7]) and *LACKI-MRAC* employs LACKI [5].

In what is to follow, we utilise our LoKI method as the adaptive element and compare it against these established alternative learning methods. Following the nomenclature of the previous methods, we name the resulting adaptive controller *LoKI-MRAC*.

2) Control of an F-4 fighter jet under wing rock: As pointed out in [9], modern fighter aircraft designs are susceptible to lightly damped oscillations in roll known as "wing rock". Commonly occurring during landing [21], 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 [8], [18], [9].

We choose to replicate the experiments of Chowdhary et al. [8], [7][1] for comparison. 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. The task was to control the aircraft's ailerons between times t_0 and t_f in 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 $\xi(\cdot)$, with the first component of the state and reference being the roll angle and the second being the angular velocity.

Since wing rock can destabilise the dynamics, the authors proposed using GP-MRAC as controller which allows to learn a model of the wing rock dynamics online. Choosing the feedback gains of the linear pseudo controller to be $K_1 = K_2 = 1$ (see [8] for more explanations), they demonstrated this could significantly improve tracking performance over competing methods.

In our experiments, we replicate their experimental setup with the only exception of setting the time increments to $0.01[s]$ instead of $0.005[s]$ to reduce overall experimental time and compare the performances of LoKI-MRAC, LACKI-MRAC and GP-MRAC. As a baseline comparison and to put our results into perspective, we also test the performance of a simple $PD-$ controller with just the feedback gains (i.e. with $\nu_{ad} = 0$).

We create 30 randomised test runs of the wing rock dynamics and tracking problem and test each control algorithm on each one of them. For each of these test runs, the initial state $x(t_0)$ is drawn uniformly from $[0, 7] \times [0, 7]$, and the kernel length scale of the GP models is drawn uniformly from $[0.05, 2]$. The covariance function of the GP models used is the squared exponential covariance function (see [20] for a detailed discuission on GP covariance functions). We follow Chowdhary et al. [7], [9] and keep the hyperparameters of the GP fixed instead of optimising it, in order to cope with the computational real time constraints of online adaption during runtime. For LoKI, we set $m = 10$. For both LACKI and LoKI, the Lipschitz constant estimate gets updated with each new sample point received. In addition, the partitioning of the input space gets updated every 100 timesteps for LoKI by reevaluating the minimal and maximal values per dimension of all observed sample points. Fig. 2 illustrates this repartitioning.

The performance of all controllers across these randomised trials is depicted in Fig. 3. 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*: average angular position error (log-deg), i.e. $\log(\frac{1}{t_f} \int_{t_0}^{t_f} \|\xi_1(t) - x_1(t)\| \, dt)$.
- *log-XDOTERR*: average roll rate error (log-deg/sec.), i.e. $\log(\frac{1}{t_f} \int_{t_0}^{t_f} \|\xi_2(t) - x_2(t)\| \, dt)$.
- *log-PREDERR*: average log-prediction error, i.e. $\log(\frac{1}{t_f} \int_{t_0}^{t_f} \left\| \hat{f}_n(x(t)) - f(x(t)) \right\| dt)$ where f is a vector field affected by the wing rock.
- *log-CMD*: average cumulative control magnitude (log-scale), i.e. $\log(\frac{1}{t_f} \int_{t_0}^{t_f} \|u(t)\| \, dt)$.
- *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.
- *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 by magnitudes in terms of tracking error. Among the adaptive controllers, the tracking errors of LoKI-MRAC seem to be slightly superior, as their outliers are lower than those of LACKI-MRAC and GP-MRAC while the bulk of tracking errors are at similar levels.

More importantly, as expected by the theory provided in the previous section, LoKI-MRAC outperforms LACKI-MRAC both with regard to training time and prediction time. It also outperforms GP-MRAC in terms of training time, while being roughly comparable in terms of prediction time. If

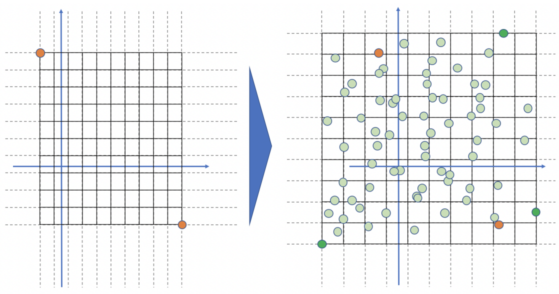

Fig. 2. Illustration of repartitioning with LoKI in online setting. Left side: initially, the first two points (orange) span the hyperrectangle H which is partitioned into m^d hyperrectangles (t = t0). Right side: after 100 time steps, with 100 new data points (green) the partitioning is redone, this time 3 of the new points (darker green) span H (t= t100). Dotted lines indicate the assignment of points to hyperrectangles beyond H as derived from previously observed sample points.

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

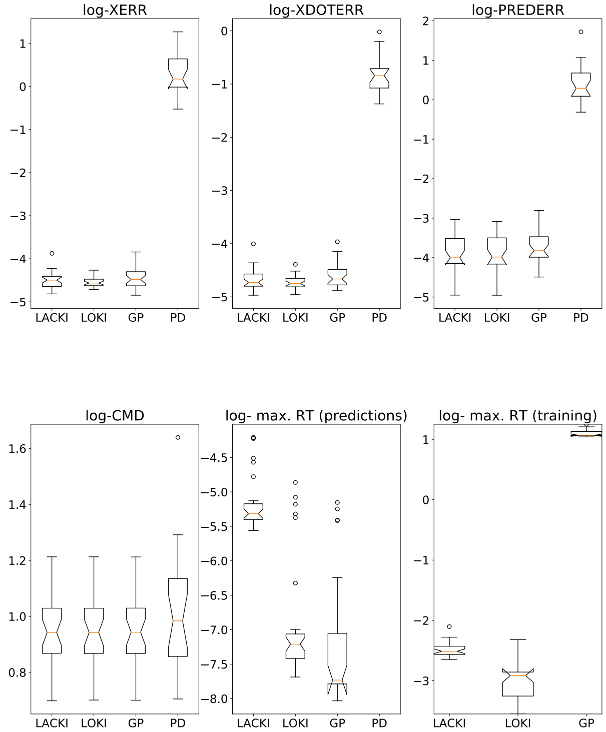

Fig. 3. Performance of the different online controllers over a range of 30 trials with randomised parameter settings and initial conditions. We see that LoKI exhibited similar or better performance than the other methods while markedly outperforming the other learning methods in terms of time required for learning.

the controller is given a finite time budget for training and prediction at each time step, as one would expect in a real world control setting, this would make LoKI-MRAC the controller of choice among the controllers tested. Overall, it can therefore be concluded that LoKI-MRAC exhibits superior performance compared to the other tested methods in our wing rock experiments and that LoKI therefore seems well-suited to be employed in control problems.

B. UCI data sets

While the wing rock dynamics control experiment indicates the advantages and suitability of LoKI for control problems, its dimensionality is rather low ($d = 2$). Also, the learning models are trained in an online setting, rather than from scratch. In order to demonstrate the computational speed-up that can be achieved with LoKI when training is done from scratch even in high-dimensional environments, we also test it on a series of machine learning benchmark regression data sets that are publicly available in the UCI data base.

Table I shows their characteristics. Each data set gets randomly split into 80% training and 20% test data and subsequently, we train LoKI, LACKI, and a range of GPs with different covariance functions (namely the automatic relevance determination version of the squared exponential (SE), the Matern32 and the Matern52 kernels) on each of the resulting training sets.

TABLE I

Characteristics of the UCI data sets. Data sets range from 308 to $45,730$ data points and from 4 to 13 dimensions.

Data Set	Data Points	Dimen- sions	m for LoKI
Boston Housing	504	13	*3*
Concrete Strength	1,030	8	4
Energy Efficiency	768	8	2
Kin8nm	8,192	8	4
Power Plant	9,568	4	9
Protein Structure	45,730	9	6
Yacht Dynamics	308	6	2

For LoKI, the we adjusted m^d, the number of hyperrectangle, according to dimensionality and size of data set in order to get both enough reduction in terms of computational complexity and still enough (expected) number of samples per sub-hypercube to get a reliable estimate of the local Lipschitz constant. The chosen m is also included in Table I. GP training was done by marginal likelihood maximisation as implemented in the GPflow package with 5 randomised restarts. For data sets with more than $2,000$ training points, we resorted to sparse approximations through variational inference as proposed in [24] using 250 inducing points. The results of our experiments are exhibited in Table II.

Discussion: As expected, LoKI is consistently the fastest method to train and is magnitudes faster than LACKI as well as standard GP training and still at least 4-5 times faster than sparse GP training. However, its prediction accuracy is often outperformed by the best prediction results obtained by GPs, which we suspect to be able to infer more structure in data (e.g. Yacht Dynamics, Energy Efficiency). Nonetheless, there are other data sets in which LoKI's prediction performance is comparable or better to the best performance obtained with GPs (e.g. Power Plant, Protein Structure). We also note that the predictive performance of LoKI on the UCI data sets is always at least on par with the (global) LACKI method, and in some cases (Energy Efficiency), where smoothness in the data seems to vary locally, even superior. Most importantly, like LACKI, LoKI training is robust in the sense that the prediction performances shown can be achieved with one single training run. As the worst case results in brackets indi-

TABLE II

Training times and test set RMSE for different UCI datasets. For GPs, best (worst) test set RMSE results obtained among all trained GPs are given; results obtained with sparse GPs are indicated by *.

	Train time in s			RMSE		
Data Set	**GP**	**LAC**	**LoKI**	**GP**	**LAC**	**LoKI**
Boston Housing	*48*	*4*	*<1*	*0.30 (1.04)*	*0.44*	*0.46*
Concrete Strength	50	20	1	0.33 (0.34)	0.57	0.58
Energy Efficiency	40	9	<1	0.04 (0.22)	0.25	0.13
Kin8nm*	131	943	1	0.29 (1.00)	0.47	0.49
Power Plant*	228	1,074	3	0.22 (0.22)	0.25	0.25
Protein Structure*	1,233	24,643	286	0.68 (0.70)	0.69	0.67
Yacht Dynamics	2	1	<1	0.02 (0.03)	0.39	0.39

cate, this is not true for GPs, which can have poor predictive performance (e.g. Boston Housing, Kin8nm) if only trained once / for one specific covariance function. In addition, for some data sets, the GP training process occasionally aborts unsuccessfully, which additionally calls for multiple training runs in order to get (any) predictive capacity. Multiple training runs can however inflate computational cost (which is already higher than LoKI) to unacceptable levels for many applications, especially in control.

V. CONCLUSIONS

We have presented LoKI, an extension of recent work on nonparametric regression related to Lipschitz interpolation and NSM methods. In contrast previous work, we proposed to estimate Lipschitz hyperparameters of the prediction model locally and in a manner that allows us to bound the expected computational effort for training as well as for prediction. We presented simulations of a learning-based model-reference adaptive control application illustrating that our approach can effectively learn a predictive dynamical system online, resulting in reliable control performance. Furthermore, our results demonstrate that, owing to its computational advantages, LoKI could operate at a much higher sample frequency as part of an online learning-based controller than competing nonparametric learning methods could. In a range of additional experiments on standard machine learning benchmark data sets, we have confirmed the computational advantages of LoKI even in higher dimensional settings and demonstrated that its predictive performance is at least on par with the global LACKI method also on more complex inference problems. In addition to its practical benefits, we provided some first worst-case consistency guarantees. Future work could readily translate online learning and control convergence guarantees, which exist for the related LACKI method, to our LoKI approach. In addition, we intend to convert our bounds on the expected computational effort into worst-case as well as into high-probability bounds and consider deployment in the context of robust data-based model-predictive control.

REFERENCES

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

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

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

[4] J. Calliess, S. J. Roberts, C. E. Rasmussen, and J. Maciejowski. Nonlinear set membership regression with adaptive hyper-parameter estimation for online learning and control. In *ECC*, 2018.

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

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

[7] 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.

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

[9] 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.

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

[11] M. P. Deisenroth, D. Fox, and C. E. Rasmussen. Gaussian processes for data-efficient learning in robotics and control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015.

[12] M. P. Deisenroth, G. Neumann, and J. Peters. A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2013.

[13] L. Gyoerfi, M. Kohler, A. Krzyzak, and H. Walk. *A Distribution-Free Theory of Nonparametric Regression*. Springer, 2002.

[14] D. Limon, J. Calliess, and Jan Marian Maciejowski. Learning-based nonlinear model predictive control. *IFAC-PapersOnLine*, 50(1):7769–7776, 2017.

[15] J. M. Manzano, D. Limon, D. Munoz de la Pena, and J. Calliess. Robust data-based model predictive control for nonlinear constrained systems. *NMPC*, 2018.

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

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

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

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

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

[21] 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.

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

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

[24] Michalis Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial Intelligence and Statistics*, pages 567–574, 2009.

[25] A. B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer, 2009.

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