2024 European Control Conference (ECC)
June 25-28, 2024. Stockholm, Sweden

Basis-functions nonlinear data-enabled predictive control: Consistent and computationally efficient formulations

M. Lazar

Abstract— **This paper considers the extension of data-enabled predictive control (DeePC) to nonlinear systems via general basis functions. Firstly, we formulate a basis-functions DeePC behavioral predictor and identify necessary and sufficient conditions for equivalence with a corresponding basis-functions multi-step identified predictor. The derived conditions yield a dynamic regularization cost function that enables a well-posed (i.e., consistent with the multi-step identified predictor) basis-functions formulation of nonlinear DeePC. Secondly, we develop two alternative, computationally efficient basis-functions DeePC formulations that use a simpler, sparse regularization cost function and ridge regression, respectively. An insightful relation between Koopman DeePC and basis-functions DeePC is also presented. The effectiveness of the developed basis-functions DeePC formulations is shown on a benchmark nonlinear pendulum state-space model, for both noise-free and noisy data, while using only output measurements.**

I. Introduction

Model predictive control (MPC) is one of the most successful advanced control methodologies due to its capability to handle constraints, optimize performance and anticipate reference changes. A prediction model can be obtained from physics or using identification. When such a model is simulated to predict future trajectories, modeling/identification errors propagate. To mitigate this, a subspace predictive control (SPC) algorithm was developed in [1] using subspace identification techniques, which yields an unbiased identified multi-step predictor and removes the need of a state estimator. More recently, a data-enabled predictive control (DeePC) algorithm was developed in [2], which utilizes a behavioral, data-based multi-step predictor. For noise-free data, equivalence of MPC and DeePC was established in [2] and equivalence of SPC and DeePC was established in [3]. In the case of noisy data, DeePC requires a regularization cost function to yield consistent predictions [4]. With the theoretical foundation of DeePC for linear systems sufficiently developed, extensions of DeePC to nonlinear systems were pursued, as most real-life systems are nonlinear.

In [5], a fundamental lemma for linear parameter varying (LPV) systems was derived, which enabled a LPV formulation of DeePC. A fundamental lemma extension for systems linear in the state, but possibly with input and/or output nonlinear terms was derived in [6], which also suggested using basis functions for learning the nonlinear terms. In [7], a data-driven predictive control algorithm for nonlinear systems was developed based on data-driven linearization along closed-loop trajectories. In [8] data-driven predictive control methods were developed for feedback linearizable nonlinear systems. In [9], reproducing kernel functions were utilized to obtain a linear lifted state-space representation of a nonlinear state-space model, accompanied with a corresponding DeePC formulation. In [10], a similar path was followed but by using a linear-in-control input Koopman lifting [11] instead. In [12], the reproducing kernel Hilbert space (RKHS) theory was utilized to construct a linear-in-the-parameters multi-step predictor of the nonlinear autoregressive exogenous (NARX) type. Then a kernelized DeePC formulation was developed therein based on the analytic closed-form kernelized NARX predictor, along with methods for robust predictions in the presence of noisy output measurements. In [13], a structured basis-functions representation of one-step NARX models was used to derive a DeePC like algorithm, which solves a tracking problem.

In this paper we consider the formulation of DeePC based on general basis-functions transformation of system trajectories. The term *general* means that we allow any class of basis functions, such as radial, polynomial or orthogonal basis functions, and we do not require structuring the basis functions in terms of past/future or input/output data. To analyse consistency of the basis-functions DeePC predictions, we construct a corresponding basis-functions subspace predictive control (SPC) problem, which utilizes unbiased multi-step NARX predictors. Then we provide necessary and sufficient conditions under which the predictions of basis-functions DeePC are consistent (i.e., in a model equivalence sense [14]) with the predictions of basis-functions SPC. Furthermore, we derive a novel dynamic regularization cost that enforces consistency of the basis-functions nonlinear DeePC predictions also in the presence of noisy data.

II. Preliminaries

Throughout this paper, for any finite number $q \in \mathbb{N}_{\geq 1}$ of column vectors or functions $\{\xi_1, \ldots, \xi_q\}$ we will make use of the operator $\mathrm{col}(\xi_1, \ldots, \xi_q) := [\xi_1^\top, \ldots, \xi_q^\top]^\top$.

We consider controllable and observable MIMO nonlinear systems with inputs $u \in \mathbb{R}^m$ and measured outputs $y \in \mathbb{R}^p$

$$\begin{aligned} x(k+1) &= \tilde{f}(x(k), u(k)), \quad k \in \mathbb{N}, \\ y(k) &= \tilde{h}(x(k)), \end{aligned} \tag{1}$$

where $x \in \mathbb{R}^n$ is an unknown state and $\tilde{f}$, $\tilde{h}$ are suitable functions. In MPC, given an initial measured or estimated state at time $k \in \mathbb{N}$, the above system equations are used to compute a sequence of predicted outputs $\mathbf{y}_{[1,N]}(k) := \{y(1|k), \ldots, y(N|k)\}$, given a sequence of predicted inputs $\mathbf{u}_{[0,N-1]}(k) := \{u(0|k), \ldots, u(N-1|k)\}$.

The author is with the Department of Electrical Engineering, Eindhoven University of Technology, The Netherlands, E-mail: m.lazar@tue.nl.

In *indirect* data-driven predictive control, multi-step NARX predictors are identified from the input/measured output data generated by system (1) and used to predict $\mathbf{y}_{[1,N]}(k)$ from past inputs and outputs and $\mathbf{u}_{[0,N-1]}(k)$, i.e.,

$$\mathbf{y}_{[1,N]}(k) := \mathbb{F}(\mathbf{u}_{\text{ini}}(k), \mathbf{y}_{\text{ini}}(k), \mathbf{u}_{[0,N-1]}(k)), \tag{2}$$

where $\mathbb{F} := \text{col}(f_1, \dots, f_N)$ and

$$\begin{aligned}\mathbf{u}_{\text{ini}}(k) &:= \text{col}(u(k - T_{\text{ini}}), \dots, u(k-1)) \in \mathbb{R}^{T_{\text{ini}}m},\\ \mathbf{y}_{\text{ini}}(k) &:= \text{col}(y(k - T_{\text{ini}} + 1), \dots, y(k)) \in \mathbb{R}^{T_{\text{ini}}p},\end{aligned}$$

and where $T_{\text{ini}} \in \mathbb{N}_{\geq 1}$ defines the order of the NARX dynamics (different orders can be used for inputs and outputs, but for simplicity we use a common order). Note that since each f_i is a MIMO predictor, it is the aggregation of several MISO predictors, i.e., $f_i = \text{col}(f_{i,1}, \dots, f_{i,p})$ where each $f_{i,j}$ predicts the j-th output, i.e., for $i = 1, \dots, N$

$$\begin{aligned}y_j(i|k) &= f_{i,j}(\mathbf{u}_{\text{ini}}(k), \mathbf{y}_{\text{ini}}(k), \mathbf{u}_{[0,N-1]}(k)),\\ y(i|k) &= \text{col}(y_1(i|k), \dots, y_p(i|k)),\end{aligned}$$

where $j = 1, \dots, p$ and p is the number of outputs. As opposed to a standard, one-step NARX prediction model, which can produce a N-step ahead prediction of the output by simulation, the *multi-step* NARX predictors (2) directly compute a N-step ahead prediction of the output by evaluating in parallel N functions $\{f_1, \dots, f_N\}$ that share the same arguments, i.e., $\{\mathbf{u}_{\text{ini}}(k), \mathbf{y}_{\text{ini}}(k), \mathbf{u}_{[0,N-1]}(k)\}$.

Next, for any $k \geq 0$ (starting time instant in the data vector) and $j \geq 1$ (length of the data vector), define

$$\begin{aligned}\bar{\mathbf{u}}(k, j) &:= \text{col}(u(k), \dots, u(k + j - 1)),\\ \bar{\mathbf{y}}(k, j) &:= \text{col}(y(k), \dots, y(k + j - 1)).\end{aligned}$$

Then we can define the Hankel matrices:

$$\begin{aligned}\mathbf{U}_p &:= \begin{bmatrix}\bar{\mathbf{u}}(0, T_{\text{ini}}) & \dots & \bar{\mathbf{u}}(T-1, T_{\text{ini}})\end{bmatrix},\\ \mathbf{Y}_p &:= \begin{bmatrix}\bar{\mathbf{y}}(1, T_{\text{ini}}) & \dots & \bar{\mathbf{y}}(T, T_{\text{ini}})\end{bmatrix},\\ \mathbf{U}_f &:= \begin{bmatrix}\bar{\mathbf{u}}(T_{\text{ini}}, N) & \dots & \bar{\mathbf{u}}(T_{\text{ini}} + T - 1, N)\end{bmatrix},\\ \mathbf{Y}_f &:= \begin{bmatrix}\bar{\mathbf{y}}(T_{\text{ini}} + 1, N) & \dots & \bar{\mathbf{y}}(T_{\text{ini}} + T, N)\end{bmatrix},\end{aligned} \tag{3}$$

where $T \geq (m + p)T_{\text{ini}} + mN$ is the number of columns of the Hankel matrices.

Then, if we parameterize the multi-step subspace predictor $\mathbb{F}(\mathbf{u}_{\text{ini}}, \mathbf{y}_{\text{ini}}, \mathbf{u}_{[0,N-1]}, \Theta)$ using a matrix of parameters Θ of suitable dimensions, we can formulate the nonlinear least squares problem:

$$\min_{\Theta} \left\| \mathbf{Y}_f - \bar{\mathbb{F}}(\mathbf{U}_p, \mathbf{Y}_p, \mathbf{U}_f, \Theta) \right\|_F^2, \tag{4}$$

where $\|\cdot\|_F$ denotes the matrix Frobenius norm and, every column of the matrix $\bar{\mathbb{F}}(\mathbf{U}_p, \mathbf{Y}_p, \mathbf{U}_f, \Theta)$ is obtained by evaluating the map $\mathbb{F}$ defined in (2) with every column of the data matrix $\begin{bmatrix}\mathbf{U}_p^\top & \mathbf{Y}_p^\top & \mathbf{U}_f^\top\end{bmatrix}^\top$ as an argument.

III. Main results

In what follows we will define a basis-functions representation of the multi-step nonlinear predictor (2). To this end, for every predicted output y_j, $j = 1, \dots, p$, consider a finite set of basis functions $\{\phi_0, \phi_1, \dots, \phi_L\}$, $L \in \mathbb{N}_{\geq 1}$ and define

$$\begin{aligned}y_j(i|k) &= \sum_{l=0}^{L} \theta_{l,j}^i \phi_l(\mathbf{u}_{\text{ini}}(k), \mathbf{y}_{\text{ini}}(k), \mathbf{u}_{[0,N-1]}(k))\\ &= \begin{bmatrix}\theta_{0,j}^i & \dots & \theta_{L,j}^i\end{bmatrix} \bar{\phi}(\mathbf{u}_{\text{ini}}(k), \mathbf{y}_{\text{ini}}(k), \mathbf{u}_{[0,N-1]}(k)),\end{aligned}$$

which corresponds to a basis-functions representation of the MISO functions $f_{i,j}$ using a common set of basis functions. Above $\bar{\phi} := \text{col}(\phi_0, \dots, \phi_L)$ and $\phi_0(\cdot) := 1$ to account for a vector of affine constant terms (biases) in (2). By stacking up all predicted outputs for all future time instants $i = 1, \dots, N$ we obtain a linear-in-the-parameters representation of the NARX multistep predictor $\mathbb{F}$, i.e.,

$$\mathbf{y}_{[1,N]}(k) := \Theta \bar{\phi}(\mathbf{u}_{\text{ini}}(k), \mathbf{y}_{\text{ini}}(k), \mathbf{u}_{[0,N-1]}(k)). \tag{5}$$

Next, define

$$\Phi := \bar{\phi}(\mathbf{U}_p, \mathbf{Y}_p, \mathbf{U}_f) \in \mathbb{R}^{(L+1)\times T} \tag{6}$$

as a matrix of data obtained by evaluating the map $\bar{\phi}$ at every column of the data matrix $\begin{bmatrix}\mathbf{U}_p^\top & \mathbf{Y}_p^\top & \mathbf{U}_f^\top\end{bmatrix}^\top$. More precisely, the element in line i, $1 \leq i \leq L + 1$ and column j, $1 \leq j \leq T$ of Φ is given by $\phi_{i-1}\left(\begin{bmatrix}\mathbf{U}_p\\ \mathbf{Y}_p\\ \mathbf{U}_f\end{bmatrix}_{:j}\right)$, where $Q_{:j}$ denotes the j-th column of any matrix Q. Since every column of the data matrix is a system trajectory, Φ represents the basis-functions transformation of these trajectories.

Then the nonlinear least squares problem (4) becomes the least squares problem:

$$\min_{\Theta} \|\mathbf{Y}_f - \Theta\Phi\|_F^2. \tag{7}$$

Assuming that the input data and the set of basis functions are such that Φ has full row-rank, we obtain the least squares optimal solution:

$$\Theta^* := \mathbf{Y}_f\Phi^\dagger = \mathbf{Y}_f\Phi^\top(\Phi\Phi^\top)^{-1}. \tag{8}$$

Next we can define the basis-functions SPC controller.

Problem III.1 (ϕ-SPC)

$$\min_{\Xi(k)} \quad l_N(y(N|k)) + \sum_{i=0}^{N-1} l_s(y(i|k), u(i|k)) \tag{9a}$$

subject to constraints:

$$\mathbf{y}_{[1,N]}(k) = \Theta^* \bar{\phi}(\mathbf{u}_{\text{ini}}(k), \mathbf{y}_{\text{ini}}(k), \mathbf{u}_{[0,N-1]}(k)) \tag{9b}$$

$$(\mathbf{y}_{[1,N]}(k), \mathbf{u}_{[0,N-1]}(k)) \in \mathbb{Y}^N \times \mathbb{U}^N \tag{9c}$$

where $\Xi(k) := \text{col}(\mathbf{y}_{[1,N]}(k), \mathbf{u}_{[0,N-1]}(k))$ are the optimization variables, $\mathbb{Y}, \mathbb{U}$ are proper polytopic sets that represent constraints, $l_s(y, u) := \|y - y_r\|_Q^2 + \|u - u_r\|_R^2$ is a stage cost and $l_N(y)$ is a terminal cost, taken for simplicity as $l_s(y, 0)$.

The basis-functions DeePC controller is defined next.

Problem III.2 (ϕ-DeePC)

$$\min_{\Xi(k)} \quad l_N(y(N|k)) + \sum_{i=0}^{N-1} l_s(y(i|k), u(i|k)) \tag{10a}$$

subject to constraints:

$$\begin{bmatrix} \Phi \\ \mathbf{Y}_f \end{bmatrix} \mathbf{g}(k) = \begin{bmatrix} \bar{\phi}(\mathbf{u}_{\text{ini}}(k), \mathbf{y}_{\text{ini}}(k), \mathbf{u}_{[0,N-1]}(k)) \\ \mathbf{y}_{[1,N]}(k) \end{bmatrix} \tag{10b}$$

$$(\mathbf{y}_{[1,N]}(k), \mathbf{u}_{[0,N-1]}(k)) \in \mathbb{Y}^N \times \mathbb{U}^N \tag{10c}$$

where $\Xi(k) := \text{col}(\mathbf{y}_{[1,N]}(k), \mathbf{u}_{[0,N-1]}(k), \mathbf{g}(k))$ are the optimization variables.

Next, we introduce a notion of system model equivalence (or consistency) inspired by [14].

Definition III.3 Two models $\{M_1, M_2\}$ of system (1) are called equivalent (or *consistent*) if for every constraints admissible input sequence $\mathbf{u}_{[0,N-1]}$ and initial condition

$$\|\mathbf{y}^{M_1}_{[1,N]} - \mathbf{y}_{[1,N]}\| = \|\mathbf{y}^{M_2}_{[1,N]} - \mathbf{y}_{[1,N]}\|, \tag{11}$$

where $\mathbf{y}_{[1,N]}$ is the true system (1) output.

Notice that one way to establish equivalence/consistency of two different models is to show that $\mathbf{y}^{M_1}_{[1,N]} = \mathbf{y}^{M_2}_{[1,N]}$. The above consistency notion (which differs from the classical consistency notion used in systems identification) is very useful for nonlinear data-driven predictive control, as in the nolinear case, the Willems' fundamental lemma [15] does not hold. Indeed, given a trustworthy, unbiased identified model, such as (5), we can use its predictions as a guiding standard for achieving consistent, i.e., equivalent, predictions in nonlinear (ϕ-)DeePC. *Hence, for want of a nonlinear fundamental lemma all is not lost, as long as equivalence with a consistent model is guaranteed, as stated next.*

Lemma III.4 *Consider the ϕ-SPC prediction model* (9b) *and the ϕ-DeePC prediction model* (10b) *defined using the same set of data $\{\mathbf{U}_p, \mathbf{Y}_p, \mathbf{U}_f, \mathbf{Y}_f\}$ generated using system* (1) *and the same set of basis functions $\{\phi_0, \phi_1, \dots \phi_L\}$. Assume that the basis-functions transformed data matrix Φ has full row-rank. Let $\mathcal{E} := \mathbf{Y}_f - \Theta^*\Phi$ be the matrix of residuals of the least squares problem* (7) *and let $\mathcal{S}_\mathbf{g} := \{\Phi^\dagger\bar{\phi}(\cdot) + \hat{\mathbf{g}} \ : \ \hat{\mathbf{g}} \in \mathcal{N}(\Phi)\} \subset \mathbb{R}^T$ be a set of parameters $\mathbf{g}$, where $\mathcal{N}(\Phi)$ is the null-space of Φ. Then the ϕ-SPC prediction model* (9b) *is equivalent with the ϕ-DeePC prediction model* (10b) if and only if $\mathcal{E}\hat{\mathbf{g}} = \mathbf{0}$ *for all* $\hat{\mathbf{g}} \in \mathcal{N}(\Phi)$.

Proof: The proof follows a similar reasoning as in the proof of Theorem 1 in [3], *mutatis mutandis*. From (10b), it follows that $\Phi\mathbf{g}(k) = \bar{\phi}(\mathbf{u}_{\text{ini}}(k), \mathbf{y}_{\text{ini}}(k), \mathbf{u}_{[0,N-1]}(k))$ and thus all variables $\mathbf{g}(k)$ that satisfy this system of equations satisfy $\mathbf{g}(k) \in \mathcal{S}_\mathbf{g} = \{\Phi^\dagger\bar{\phi}(\cdot) + \hat{\mathbf{g}} \ : \ \hat{\mathbf{g}} \in \mathcal{N}(\Phi)\}$. Therefore, predicted outputs generated by ϕ-DeePC satisfy

$$\mathbf{y}_{[1,N]}(k) \in \{\mathbf{Y}_f\Phi^\dagger\bar{\phi}(\cdot) + \mathbf{Y}_f\hat{\mathbf{g}} \ : \ \hat{\mathbf{g}} \in \mathcal{N}(\Phi)\}.$$

Based on (7)-(8) and $\hat{\mathbf{g}} \in \mathcal{N}(\Phi)$, we have that

$$\mathbf{Y}_f\hat{\mathbf{g}} = (\mathcal{E} + \Theta^*\Phi)\,\hat{\mathbf{g}} = \mathcal{E}\hat{\mathbf{g}} + \Theta^*\Phi\hat{\mathbf{g}} = \mathcal{E}\hat{\mathbf{g}}.$$

Thus, it holds that

$$\mathbf{y}^{\phi-\text{DeePC}}_{[1,N]}(k) = \mathbf{Y}_f\Phi^\dagger\bar{\phi}(\cdot) = \Theta^*\bar{\phi}(\cdot) = \mathbf{y}^{\phi-\text{SPC}}_{[1,N]}$$

if and only if $\mathcal{E}\hat{\mathbf{g}} = \mathbf{0}$ for all $\hat{\mathbf{g}} \in \mathcal{N}(\Phi)$. ■

In the deterministic, noise-free, linear case, the above result recovers the result of Theorem 1 in [3] because in the linear case the outputs can be exactly predicted from finite persistently exciting data, i.e., $\mathcal{E} = \mathbf{0}$. However, the proof of Lemma III.4 shows that in the case of noise-free data generated by a nonlinear system (or in the case of *noisy data* generated by a linear or nonlinear system) the ϕ-DeePC predictor will not necessarily be consistent with the unbiased ϕ-SPC predictor. Hence, for nonlinear systems, even in the deterministic, noise-free, case a regularization cost is required to enforce consistent predictions. Alternatively, the basis functions should be such that the nonlinear system outputs can be exactly predicted from finite data.

Hence, since in general in the deterministic nonlinear case or the noisy data case $\mathcal{E} \neq \mathbf{0}$, the remaining option to achieve consistent predictions in ϕ-DeePC is to regularize the variables $\mathbf{g}$ such that $\mathcal{S}_\mathbf{g} \approx \{\Phi^\dagger\bar{\phi}(\cdot)\}$ or, alternatively, to regularize the variables $\hat{\mathbf{g}}$ to zero. The first option yields the following regularized basis function DeePC controller.

Problem III.5 (ϕ-DeePC-R1)

$$\min_{\Xi(k)} \quad l_N(y(N|k)) + \sum_{i=0}^{N-1} l_s(y(i|k), u(i|k)) + l^1_g(\mathbf{g}(k)) \tag{12a}$$

subject to constraints:

$$\begin{bmatrix} \Phi \\ \mathbf{Y}_f \end{bmatrix} \mathbf{g}(k) = \begin{bmatrix} \bar{\phi}(\mathbf{u}_{\text{ini}}(k), \mathbf{y}_{\text{ini}}(k), \mathbf{u}_{[0,N-1]}(k)) \\ \mathbf{y}_{[1,N]}(k) \end{bmatrix} \tag{12b}$$

$$(\mathbf{y}_{[1,N]}(k), \mathbf{u}_{[0,N-1]}(k)) \in \mathbb{Y}^N \times \mathbb{U}^N \tag{12c}$$

where

$$l^1_g(\mathbf{g}(k)) := \lambda\|\mathbf{g}(k) - \mathbf{g}_r(k)\|^2_2, \tag{13}$$

and $\mathbf{g}_r(k) := \Phi^\dagger\bar{\phi}(\mathbf{u}_{\text{ini}}(k), \mathbf{y}_{\text{ini}}(k), \mathbf{u}_{[0,N-1]}(k))$. The second option above yields the following ϕ-DeePC controller.

Problem III.6 (ϕ-DeePC-R2)

$$\min_{\Xi(k)} \quad l_N(y(N|k)) + \sum_{i=0}^{N-1} l_s(y(i|k), u(i|k)) + l^2_g(\hat{\mathbf{g}}(k)) \tag{14a}$$

subject to constraints:

$$\Phi\hat{\mathbf{g}}(k) = 0 \tag{14b}$$

$$\mathbf{Y}_f\left(\Phi^\dagger\bar{\phi}(\mathbf{u}_{\text{ini}}(k), \mathbf{y}_{\text{ini}}(k), \mathbf{u}_{[0,N-1]}(k)) + \hat{\mathbf{g}}(k)\right) = \mathbf{y}_{[1,N]}(k) \tag{14c}$$

$$(\mathbf{y}_{[1,N]}(k), \mathbf{u}_{[0,N-1]}(k)) \in \mathbb{Y}^N \times \mathbb{U}^N \tag{14d}$$

where

$$l^2_g(\hat{\mathbf{g}}(k)) := \lambda\|\hat{\mathbf{g}}(k)\|^2_2 \tag{15}$$

and $\Xi(k) := \text{col}(\mathbf{y}_{[1,N]}(k), \mathbf{u}_{[0,N-1]}(k), \hat{\mathbf{g}}(k))$ are the optimization variables. The novel regularization cost (13) is

called dynamic because $\mathbf{g}_r(k)$ changes at every time instant k. Notice that this is different from the typical regularization costs used in linear DeePC [4], e.g., $\|\mathbf{g}(k)\|_2^2$ (does not yield a consistent DeePC predictor) or $\|(I-\Pi)\mathbf{g}(k)\|_2^2$ (yields a consistent DeePC predictor) where Π is a constant data matrix. Compared to the Π-regularization cost [4], the developed dynamic cost is computationally more efficient while achieving consistency with the SPC predictor.

The proof of Lemma III.4 shows that $\mathbf{y}_{[1,N]}^{\phi-\text{DeePC}-R1}(k) = \mathbf{y}_{[1,N]}^{\phi-\text{DeePC}-R2}(k)$ for the same cost function weighting matrices and λ parameter, i.e., since $\mathbf{g}(k)-\mathbf{g}_r(k) = \hat{\mathbf{g}}(k)$ for some $\hat{\mathbf{g}}(k) \in \mathcal{N}(\Phi)$. However, Problem III.6 is computationally more efficient than Problem III.5 as it uses a static quadratic cost with a sparse Hessian.

The result of Lemma III.4 only assumes that the matrix Φ has full row-rank, i.e., it does not assume noise-free data. Hence, the ϕ-SPC equivalence conditions of Lemma III.4 also yield a consistent regularization method for noisy output data. Moreover, the ϕ-DeePC-R2 predictor equations (14b)-(14c) always admit the solution $\hat{\mathbf{g}}(k) = \mathbf{0}$ even when the output measurements are affected by noise, case in which the ϕ-DeePC-R2 predictor reduces to the unbiased least squares optimal ϕ-SPC predictor.

If the matrix Φ does not have full row-rank, a ridge regression solution, as employed in [12] for kernelized DeePC, can be computed as

$$\Theta^{R*} := \mathbf{Y}_f \Phi^\top (\Phi\Phi^\top + \gamma I)^{-1} \tag{16}$$

where γ is a positive scalar and I is an identity matrix of suitable dimensions. This yields the next formulation.

Problem III.7 (Ridge ϕ-DeePC)

$$\min_{\Xi(k)} \quad l_N(y(N|k)) + \sum_{i=0}^{N-1} l_s(y(i|k), u(i|k)) \tag{17a}$$

subject to constraints:

$$\begin{bmatrix} \Phi\Phi^\top + \gamma I \\ \mathbf{Y}_f \Phi^\top \end{bmatrix} \mathbf{g}(k) = \begin{bmatrix} \bar{\phi}(\mathbf{u}_{\text{ini}}(k), \mathbf{y}_{\text{ini}}(k), \mathbf{u}_{[0,N-1]}(k)) \\ \mathbf{y}_{[1,N]}(k) \end{bmatrix} \tag{17b}$$

$$(\mathbf{y}_{[1,N]}(k), \mathbf{u}_{[0,N-1]}(k)) \in \mathbb{Y}^N \times \mathbb{U}^N. \tag{17c}$$

For a proof of consistency of the ridge ϕ-DeePC formulation we refer to [16]. This formulation offers less flexibility to optimize the bias variance trade-off compared to the ϕ-DeePC-R2 formulation, but it can handle Φ matrices without a full row-rank and it offers more flexibility to reduce computational complexity. Indeed, notice that in ridge ϕ-DeePC the dimension of the vector of variables $\mathbf{g}(k) \in \mathbb{R}^{L+1}$ is dictated by the number of basis functions $L+1$ versus the data size T, as for ϕ-DeePC(-R1,-R2). This is not the case however for the kernelized DeePC in [12], which obtains $\Phi \in \mathbb{R}^{T\times T}$ as a Gram matrix, and thus, $\mathbf{g}(k) \in \mathbb{R}^T$. Hence, allowing for a non-square, fat matrix Φ in the ridge ϕ-DeePC formulation provides a useful alternative to [12].

A. Relation with Koopman MPC

Koopman model predictive control was developed in [11] as a method to apply linear MPC techniques to nonlinear systems. To this end, the idea is to lift the state $x(k)$ of the original system (1) to a higher dimensional space where the dynamics are linear, via a set of observables, which can be parameterized using basis functions, i.e.,

$$z(k) := \bar{\phi}_K(x(k)) := \text{col}(\phi_{1,K}(x(k)), \dots, \phi_{L,K}(x(k))).$$

This can also be done based on input-output data as presented in [17], which yields the following linear-in-control input embedding of the nonlinear system (1):

$$\begin{aligned} z(k+1) &= Az(k) + Bu(k), \quad k \in \mathbb{N}, \\ y(k) &= Cz(k), \\ z(0) &= \bar{\phi}_K(\mathbf{u}_{\text{ini}}(0), \mathbf{y}_{\text{ini}}(0)). \end{aligned} \tag{18}$$

The above model can be used to define

$$\Psi := \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^N \end{bmatrix}, \quad \Gamma := \begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{N-1}B & CA^{N-2}B & \cdots & CB \end{bmatrix},$$

which yields the following Koopman MPC controller.

Problem III.8 (Koopman MPC)

$$\min_{\Xi(k)} \quad l_N(y(N|k)) + \sum_{i=0}^{N-1} l_s(y(i|k), u(i|k)) \tag{19a}$$

subject to constraints:

$$\mathbf{y}_{[1,N]}(k) = \Psi z(0|k) + \Gamma \mathbf{u}_{[0,N-1]}(k) \tag{19b}$$

$$z(0|k) = \bar{\phi}_K(\mathbf{u}_{\text{ini}}(k), \mathbf{y}_{\text{ini}}(k)) \tag{19c}$$

$$(\mathbf{y}_{[1,N]}(k), \mathbf{u}_{[0,N-1](k)}(k)) \in \mathbb{Y}^N \times \mathbb{U}^N, \tag{19d}$$

where $\Xi(k) := \text{col}(\mathbf{y}_{[1,N]}(k), \mathbf{u}_{[0,N-1]}(k))$ are the optimization variables. Then, since the lifted model (18) is linear, the following result is a consequence of the results in [2], [3].

Corollary III.9 *Assume that the Koopman lifted state-space system* (18) *is controllable and observable. Assume that a persistently exciting input is used to generate noise-free output data for system* (18) *such that the matrix* $\Phi_K := \bar{\phi}(\mathbf{U}_p, \mathbf{Y}_p, \mathbf{U}_f)$ *has full row-rank, where*

$$\bar{\phi}(\mathbf{u}_{ini}, \mathbf{y}_{ini}, \mathbf{u}_{[0,N-1]}) := \begin{bmatrix} \bar{\phi}_K(\mathbf{u}_{ini}, \mathbf{y}_{ini}) \\ \mathbf{u}_{[0,N-1]} \end{bmatrix} = \begin{bmatrix} z \\ \mathbf{u}_{[0,N-1]} \end{bmatrix}. \tag{20}$$

Consider the ϕ-SPC prediction model (9b) *and the ϕ-DeePC prediction model* (10b) *defined using the same set of data* $\{\mathbf{U}_p, \mathbf{Y}_p, \mathbf{U}_f, \mathbf{Y}_f\}$ *and the same set of basis functions* $\{\phi_{1,K}, \dots \phi_{L,K}\}$ *and* $\bar{\phi}(\cdot)$ *defined as in* (20). *Then the Koopman MPC prediction model* (19b)-(19c), *the ϕ-SPC prediction model* (9b) *and the ϕ-DeePC prediction model* (10b) *are equivalent.*

The above result shows that Koopman DeePC is a special case of basis-functions DeePC, i.e., corresponding to basis

functions that are linear in the present and future control inputs. This implies that via the regularized ϕ-DeePC formulations (R1, R2 or Ridge) developed in this paper, we can obtain consistent Koopman DeePC formulations without separating the identification problem from the prediction/control synthesis problem, as proposed in [10], while still solving a single QP online.

For various choices of the basis-functions representations we refer the interested reader to [16]. For stability analysis of the developed basis-functions nonlinear DeePC controllers we refer to the dissipativity-based approach put forward in [18], which considers general nonlinear Hankel operators.

IV. Illustrative Example

Consider the following state-space model obtained via Euler discretization from the pendulum model in [19]:

$$\begin{aligned}\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} &= \begin{bmatrix} 1-\frac{bT_s}{J} & 0 \\ T_s & 1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \begin{bmatrix} \frac{T_s}{J} \\ 0 \end{bmatrix} u(k) \\ &\quad - \begin{bmatrix} \frac{MLgT_s}{2J} \sin(x_2(k)) \\ 0 \end{bmatrix} \\ y(k) &= x_2(k) + w(k),\end{aligned} \tag{21}$$

where $u(k) \in [-3, 3]$ and $y(k)$ are the system input torque and pendulum angle at time instant k, while $J = \frac{ML^2}{3}$, $M = 1$ kg and $L = 1$ m are the moment of inertia, mass and length of the pendulum. Moreover, $g = 9.81$ m/s^2 is the gravitational acceleration, $b = 0.1$ is the friction coefficient and the sampling time $T_s = \frac{1}{30}$ s. The performance of the developed basis-functions DeePC and SPC controllers is evaluated for a prediction horizon $N = 10$, $Q = 200$ and $R = 0.5$ for all algorithms. Except for one simulation when multiple λ values are specified in Figure 1, for both ϕ-DeePC-R1 and -R2 we used $\lambda = 1e+4 = 10^4$; for Ridge ϕ-DeePC we used $\gamma = 1e-3 = 10^{-3}$.

To generate the output data an open-loop identification experiment was performed using a multisine input constructed with the Matlab function *idinput*, with the parameters *Range* $[-4, 4]$, *Band* $[0, 1]$, *Period* 1000, *NumPeriod* 1 and *Sine* $[25, 40, 1]$. The data length is 1000 and $T_{\text{ini}} = 5$ is used, as estimated in [19]. For identification from noisy data white noise was added to the output with standard deviation 0.01.

We have used basis functions that are linear in the present and future inputs, as defined in (20). This allows us to solve all predictive control formulations developed in this paper via QP, using the *quadprog* solver, which provides optimal solutions, so the comparison of the obtained results is not hindered by local minima. To generate the basis functions $\phi_K(\mathbf{u}_{\text{ini}}, \mathbf{y}_{\text{ini}})$ we utilized a radial-basis-functions neural network with 30 centers/neurons with Gaussian activation functions. A hybrid neural network was created by adding the 10 linear inputs $\mathbf{u}_{[0,N-1]}$ to the 30 outputs of the Gaussian neurons in *PyTorch* and then it was trained to find the optimal centers using the *Adam* optimizer with the *MSE* loss function, the *learning rate* 0.0005 and L_2 regularization using the weight 10^{-7}. This yields 30 basis functions $\phi_{K,l}(z) := e^{-\|z-z^*_{c,l}\|_2^2}$, where $z = \text{col}(\mathbf{u}_{\text{ini}}, \mathbf{y}_{\text{ini}})$, $l = 1, \ldots, 30$. By letting $\bar{\phi}(\mathbf{u}_{\text{ini}}, \mathbf{y}_{\text{ini}}, \mathbf{u}_{[0,N-1]}) = \text{col}(\phi_{K,1}(z), \ldots, \phi_{K,30}(z), \mathbf{u}_{[0,N-1]})$ we obtain a $\Phi \in \mathbb{R}^{40\times 990}$ matrix, since $N = 10$, the data length is 1000 and we used Hankel matrices to generate $(\mathbf{U}_p, \mathbf{Y}_p, \mathbf{U}_f)$.

Formulation	J_{ISE}	J_{IAE}	J_u	J_{track}	CPU
ϕ-SPC	0.0428	0.0840	2.0796	11.1387	0.0087
ϕ-DeePC-R1	0.0429	0.0841	2.0758	11.1373	0.0955
ϕ-DeePC-R2	0.0429	0.0841	2.0758	11.1373	0.0806
Ridge ϕ-DeePC	0.0429	0.0850	2.0773	11.1469	0.0083

TABLE I: Performance & mean CPU time for noiseless data.

Firstly, we consider the noise-free data case and we test the consistency of the ϕ-DeePC-R1 formulation for a small value of the regularization weight $\lambda = 1e-1 = 0.1$ versus a large value $\lambda = 1e+4 = 10^4$. The results are shown in Figure 1 for tracking a sinusoidal reference with the frequency of 1Hz, duration of 4 seconds and 100 samples per second, i.e. $t = 0 : 0.01 : 4$, $r = \sin(2\pi Ft)$, which is used in all simulations. The initial state for the simulations is $x_1(0) = 1.1$ and $x_2(0) = 1$.

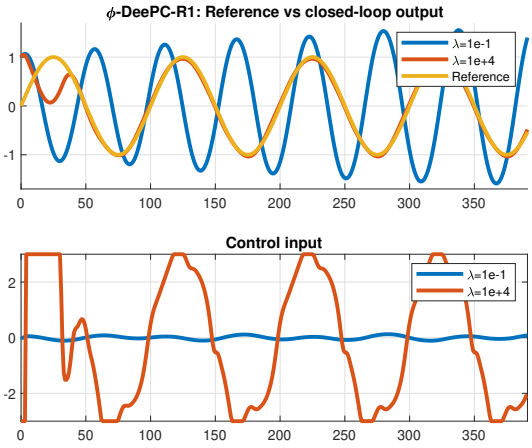

Fig. 1: ϕ-DeePC-R1: tracking performance, noise-free data.

We observe that for the small value of λ, the predictions of ϕ-DeePC are not consistent as expected, i.e., the resulting input is close to zero and the output is far from the reference, while the predictive controller estimates that such an input should give good results. However, once the variable λ is sufficiently large, we obtain consistent predictions and good tracking performance, as indicated by Lemma III.4.

To compare the performance and computational complexity of all the derived data-driven predictive controllers we report the following performance indices in Table I: $J_{ISE} = \frac{1}{T_{sim}} \sum_{k=1}^{T_{\text{sim}}} \|y(k) - r(k)\|_2^2$, $J_{IAE} = \frac{1}{T_{sim}} \sum_{k=1}^{T_{\text{sim}}} \|y(k) - r(k)\|_1$, $J_u = \frac{1}{T_{sim}} \sum_{k=1}^{T_{\text{sim}}} \|u(k)\|_1$ and $J_{track} = \frac{1}{T_{sim}} (\sum_{k=1}^{T_{sim}} \|Q^{\frac{1}{2}}(y(k) - y_r(k))\|_2^2 + \|R^{\frac{1}{2}}(u(k) - u_r(k))\|_2^2)$. The mean CPU time in seconds is also given.

We observe that as expected the ϕ-SPC and Ridge ϕ-DeePC formulations are computationally much more efficient, while the ϕ-DeePC-R1 formulation yields equivalent performance with ϕ-DeePC-R2 and both yield slightly better

Formulation	J_{ISE}	J_{IAE}	J_u	J_{track}	CPU
ϕ-SPC	0.0474	0.0979	2.2209	12.2987	0.0084
ϕ-DeePC-R1	0.0466	0.0932	2.1642	12.0210	0.1013
ϕ-DeePC-R2	0.0466	0.0932	2.1642	12.0210	0.0852
Ridge ϕ-DeePC	0.0467	0.0982	2.1815	12.0709	0.0090

TABLE II: Performance & mean CPU time for noisy data.

tracking performance overall. This is consistent with the behavior of linear DeePC, which by optimizing the variance-bias trade-off in the noisy data case, can obtain better performance than linear SPC. In this case, although there is no measurement noise, since the residuals of the ϕ-SPC predictor are not equal to zero, there is a prediction error and ϕ-DeePC-R1(-R2) can better compensate for it.

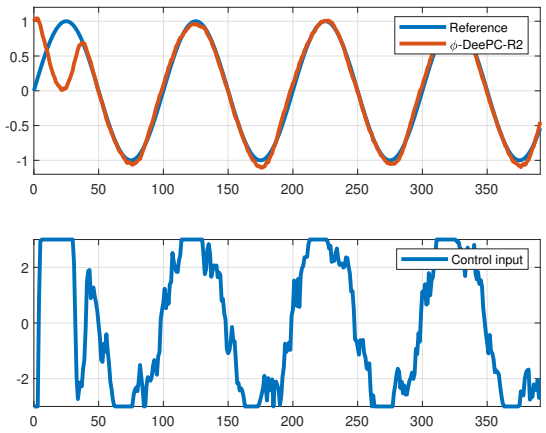

Fig. 2: ϕ-DeePC-R2: tracking performance, noisy data.

Next, we show the tracking performance for noisy data of the ϕ-DeePC-R2 in Figure 2. In this simulation also the measured output used for feedback is affected by white noise with the same standard deviation of 0.01. As guaranteed by Lemma III.4, ϕ-DeePC-R2 is robust to noisy data and results in very good tracking performance, despite a non-negligible measurement noise standard deviation. The resulting performance indicators are reported for all data-driven predictive controllers in Table II along with mean CPU times.

In both Table I and Table II the reported performances indices are divided by the simulation time (i.e., 391 *sampling instances) to obtain normalized values.* Hence, even a small difference indicates a notable improvement. E.g., if we multiply the difference in tracking error between ϕ-SPC and ϕ-DeePC-R1(R2) with 391, we obtain a total improvement in tracking error over the complete simulation by 0.5474 for the noise-free data and 108.5807 for the noisy data.

V. Conclusions

In this paper we provided a basis-functions formulation of nonlinear data-enabled predictive control. We have presented necessary and sufficient conditions for consistency of basis-functions behavioral multi-step predictors in relation with basis-functions identified multi-step predictors. From these conditions we derived two novel regularized formulations of basis-functions DeePC with guaranteed consistent predictions for both noise-free and noisy data. The consistency result in Lemma III.4 opens the door to using a wide range of powerful machine learning methods for data-enabled predictive control of nonlinear systems, which is very appealing for real-life applications.

Acknowledgements: The assistance provided by MSc. Mihai-Serban Popescu with building the hybrid radial basis-functions neural networks in PyTorch and with preparing the ϕ-SPC Matlab simulation files is gratefully acknowledged.

References

[1] W. Favoreel, B. De Moor, and M. Gevers, "SPC: Subspace predictive control," in *Proc. of the 14th IFAC World Congress*, vol. 32, no. 2. Elsevier, 1999, pp. 4004–4009.

[2] J. Coulson, J. Lygeros, and F. Dörfler, "Data-Enabled Predictive Control: In the Shallows of the DeePC," in *18th European Control Conference*, Napoli, Italy, 2019, pp. 307–312.

[3] F. Fiedler and S. Lucia, "On the relationship between data–enabled predictive control and subspace predictive control," in *IEEE Proc. of the European Control Conference (ECC)*, Rotterdam, The Netherlands, 2021, pp. 222–229.

[4] F. Dörfler, J. Coulson, and I. Markovsky, "Bridging direct indirect data-driven control formulations via regularizations and relaxations," *IEEE Transactions on Automatic Control*, 2022.

[5] C. Verhoek, H. S. Abbas, R. Tóth, and S. Haesaert, "Data-driven predictive control for linear parameter-varying systems," *IFAC-PapersOnLine*, vol. 54, no. 8, pp. 101–108, 2021, 4th IFAC Workshop on Linear Parameter Varying Systems LPVS 2021.

[6] J. Berberich and F. Allgöwer, "A trajectory-based framework for data-driven system analysis and control," in *19th European Control Conference*, Saint Petersburg, Russia, 2020, pp. 1365–1370.

[7] J. Berberich, J. Köhler, M. A. Müller, and F. Allgöwer, "Linear Tracking MPC for Nonlinear Systems—Part II: The Data-Driven Case," *IEEE Transactions on Automatic Control*, vol. 67, no. 9, pp. 4406–4421, 2022.

[8] M. Alsalti, V. G. Lopez, J. Berberich, F. Allgöwer, and M. A. Müller, "Data-driven nonlinear predictive control for feedback linearizable systems," *arXiv*, vol. 2211.06339, 2023.

[9] Y. Lian and C. N. Jones, "Nonlinear data-enabled prediction and control," in *Proceedings of the 3rd Conference on Learning for Dynamics and Control*, ser. Proceedings of Machine Learning Research, vol. 144. PMLR, 2021, pp. 523–534.

[10] Y. Lian, R. Wang, and C. N. Jones, "Koopman based data-driven predictive control," *arXiv*, vol. 2102.05122, 2021.

[11] M. Korda and I. Mezić, "Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control," *Automatica*, vol. 93, pp. 149–160, 2018.

[12] L. Huang, J. Lygeros, and F. Dörfler, "Robust and kernelized data-enabled predictive control for nonlinear systems," *arXiv*, vol. 2206.01866, 2022.

[13] A. Fazzi and A. Chiuso, "Data-driven prediction and control for NARX systems," *arXiv*, vol. 2304.02930, 2023.

[14] P. Eykhoff, *System identification : parameter and state estimation*. Wiley-Interscience, 1974.

[15] J. C. Willems, P. Rapisarda, I. Markovsky, and B. L. De Moor, "A note on persistency of excitation," *Systems and Control Letters*, vol. 54, no. 4, pp. 325–329, 2005.

[16] M. Lazar, "Basis-functions nonlinear data-enabled predictive control: Consistent and computationally efficient formulations," *arXiv*, vol. 2311.05360, 2023.

[17] M. Korda and I. Mezić, *Koopman Model Predictive Control of Nonlinear Dynamical Systems*. Cham: Springer International Publishing, 2020, pp. 235–255.

[18] M. Lazar, "A dissipativity-based framework for analyzing stability of predictive controllers," in *IFAC PapersOnLine 54–6, 7th IFAC Conference on Nonlinear Model Predictive Control*, Bratislava, Slovakia, 2021, pp. 159–165.

[19] A. Dalla Libera and G. Pillonetto, "Deep prediction networks," *Neurocomputing*, vol. 469, pp. 321–329, 2022.