Kernel-based State-Space Kriging for Predictive Control

A. Daniel Carnerero, Daniel R. Ramirez, Daniel Limon, Teodoro Alamo

***Abstract*—In this paper, we extend the State-Space Kriging (SSK) modeling technique presented in a previous work by the authors in order to consider non-autonomous systems. SSK is a data-driven method that computes predictions as linear combinations of past outputs. To model the nonlinear dynamics of the system, we propose the Kernel-based State-Space Kriging (K-SSK), a new version of the SSK where kernel functions are used instead of resorting to considerations about the locality of the data. Also, a Kalman filter can be used to improve the predictions at each time step in the case of noisy measurements. A constrained tracking Nonlinear Model Predictive Control (NMPC) scheme using the black-box input-output model obtained by means of the K-SSK prediction method is proposed. Finally, a simulation example and a real experiment are provided in order to assess the performance of the proposed controller.**

I. Introduction

THE problem of obtaining a process model from experimental data is very relevant and has implications for many control applications. Many techniques have been proposed for this task. Classic identification methods like least squares identification or the subspace method have been used since the 80s [1]. Recently, machine learning modeling techniques such as Deep Neural Networks (DNN) [2], [3], Gaussian Processes (GP) [4] or Dynamic Mode Decomposition (DMD) [5] have gained attention thanks to their ability to model the nonlinearities of a dynamic system. Other data-based techniques like Direct Weight Optimization (DWO) [6], [7] or Hölder Inference (HI) [8] have been successfully applied to modeling in control problems.

Here, we consider the State-Space Kriging (SSK) method [9]. In SSK, a model of the system is built using the vector of optimal weights obtained from a convex optimization problem as the process state. The optimization problem is closely related to that of kriging interpolation [10]. The kriging method can be used to compute predictions of a dynamic system as a linear combination of past outputs [11]. However, the problem is usually considered static and thus the past vector of weights, which might provide useful information, is not taken into account. In contrast to this, in the SSK method, dynamics are taken into account and thus a model of the system is obtained instead of a simple predictor. A limitation of the SSK method is that, in [9], the model obtained is linear, and although it can be turned into Linear-Time-Varying (LTV) considering local data, it may not be the best choice to model nonlinear systems.

Departamento de Ingeniería de Sistemas y Automática, Universidad de Sevilla, Escuela Superior de Ingenieros, Camino de los Descubrimientos s/n, 41020 Sevilla, Spain (e-mail: acarnerero,danirr,dlm,talamo@us.es). A preliminary version of this paper was presented at the 2022 IEEE Conference on Decision and Control.

This work was supported by the Agencia Estatal de Investigación (AEI)-Spain under Grant PID2019-106212RB-C41/AEI/10.13039/501100011033 and also by Junta de Andalucía and FEDER funds under grant P20_00546.

Model Predictive Control (MPC) [12] is a control scheme where a mathematical model of the system is used to compute an optimal sequence of control actions to steer the state to the desired reference. Also, it is one of the few control techniques that allows non-real-valued inputs [13] and handling constraints [14]. However, the main drawback is the necessity to obtain a good model of the process. First-principles models can be used only in very limited situations when the dynamics of the process are previously known, which can be considered as a strong assumption. For this reason, black-box or grey-box models are used more frequently [1]. In recent times, data-driven approaches to predictive control are being researched as well [15]. Also, thanks to the massive spread of Artificial Intelligence (AI) approaches, more data-driven and learning-based approaches are being considered in many different control problems [16], [17].

One of the problems in standard constrained MPC formulations is that recursive feasibility can be lost when the reference changes if the controller is not well designed. In order to solve this problem, a Tracking MPC formulation was proposed in [18], [19]. This approach preserves stability and guarantees that the system can be steered to any admissible equilibrium point, no matter what the initial equilibrium point is, without losing recursive feasibility. This strategy also provides a larger domain of attraction for a given prediction horizon N_p and, in the case of unreachable references, asymptotic convergence to the closest reachable reference is proven as well.

In this paper, we extend the results of [9] so that non-autonomous systems can be considered. Moreover, in order to improve modeling capabilities, we consider kernel functions. Kernel methods [20], [21] have become a popular technique to deal with nonlinearities because of their capability to map nonlinearly separable data into linearly separable data in a Hilbert feature space by means of the so-called kernel trick. In this work, the use of kernel functions yields a truly nonlinear SSK model, improving the Linear Time Variant (LTV) model presented in [9]. When only noisy measurements are available, a Kalman filter can be used to improve the quality of the predictions of the SSK system, as shown in the paper. The modeling strategy proposed in this paper is then used in a constrained predictive control strategy in which only the measured outputs are available. Finally, the proposed controller is tested using a numerical example and a multivariable temperature control laboratory process.

In summary, the contributions of this paper are the following:

1) We propose a version of the SSK technique that allows us to model non-autonomous systems. As the original version only considered autonomous systems, it was suitable for time-series forecasting but not in a control context. As shown in the paper, the proposed technique

can be used in control applications, obtaining good results when used within a model predictive control strategy.

2) Also, the nonlinearities of the system are modeled by means of kernel functions, unlike the previous SSK, which relied on local data akin to an online linearization. One of the advantages of the proposed kernel formulation to model nonlinear systems in comparison to the local-data approach presented in [9] is the improvement related to the computational burden and the prediction errors.
3) Unlike most of the NMPC approaches, the resulting optimization problem of the NMPC controller is a quadratic programming problem that can be easily solved even in platforms with less computing capabilities. Thus, the computational complexity of the NMPC controller is in the same class as that of a standard linear MPC. This improvement is a direct consequence of the proposed kernel formulation.

The paper is organized as follows: in section II, the methodology to obtain the system model based on SSK is presented. Section III shows how to introduce kernels to model nonlinearities, whereas section IV shows the application of the Kalman filter to the obtained system in order to tackle noisy data. Section V presents a forecasting example to show the advantages of the proposed formulation with respect to the scheme presented in [9]. In section VI, the tracking MPC formulation is introduced. Finally, application examples are shown in section VII in order to prove the effectiveness of the proposed approach.

II. State-Space Kriging for non-autonomous systems

This section presents the SSK approach for non-autonomous systems. The idea behind the SSK is to obtain an equivalent model of the process in which the state vector is composed by the weights obtained from a kriging-like interpolation. Consider a non-autonomous discrete time nonlinear system

$$x_{k+1} = f(x_k, u_k) \tag{1a}$$
$$y_k = g(x_k), \tag{1b}$$

where k is the time instant, $x_k \in \mathcal{R}^{n_x}$ is the state of the system, $u_k \in \mathcal{R}^{n_u}$ is the input of the system, $y_k \in \mathcal{R}^{n_y}$ is the output of the system, $f(\cdot)$ and $g(\cdot)$ are unknown nonlinear functions such that $f(\cdot) : \mathcal{R}^{n_x \times n_u} \rightarrow \mathcal{R}^{n_x}$ and $g(\cdot) : \mathcal{R}^{n_x} \rightarrow \mathcal{R}^{n_y}$. We will assume that the only available data are the measurable inputs and outputs. We define a time delay embedding vector $z_k \in \mathcal{R}^{n_z}$ containing the n_p past outputs of the system. That is, $z_k = [y_k^\top, y_{k-1}^\top, ..., y_{k-n_p+1}^\top]^\top \in \mathcal{R}^{n_z}$, where $n_z = n_p n_y$. We also denote z_k^+ as the successor of z_k, i.e. $z_k^+ = [y_{k+1}^\top, y_k^\top, ..., y_{k-n_p+2}^\top]^\top \in \mathcal{R}^{n_z}$. From now on, assume that some historical data of the plant is stored in a database in the form of matrices:

$$D = \left[\begin{array}{cccc} \bar{z}_1 & \bar{z}_2 & \dots & \bar{z}_N \end{array} \right],$$
$$D^+ = \left[\begin{array}{cccc} \bar{z}_1^+ & \bar{z}_2^+ & \dots & \bar{z}_N^+ \end{array} \right],$$
$$U = \left[\begin{array}{cccc} \bar{u}_1 & \bar{u}_2 & \dots & \bar{u}_N \end{array} \right],$$

where N is the number of data points, $U \in \mathcal{R}^{n_u \times N}$ is the data base of control actions, $D \in \mathcal{R}^{n_z \times N}$ contains past samples of the time-delay embedding and $D^+ \in \mathcal{R}^{n_z \times N}$ is the matrix successor of D. Note that $\bar{z}_i$ refers to a past time delay embedding and $\bar{u}_i$ refers to the past input of the system associated to $\bar{z}_i$, which resulted in the successor $\bar{z}_i^+$. Note, however, that the indexes of the columns of D, D^+ and U do not refer to the sample time, but to the position in the matrix. Therefore, $\bar{z}_{i+1}$ is not necessarily the successor sample of $\bar{z}_i$.

At sample time k and using the previously defined matrices, it is possible to compute a prediction of the successor of z_k, denoted as $\tilde{z}_{k+1}$. We could compute such prediction using a linear combination of the columns of matrix D^+ weighted by a vector of optimal weights $\mu_k^* \in \mathcal{R}^N$, i.e. $\tilde{z}_{k+1} = D^+ \mu_k^*$. This vector of weights would be computed by solving the following optimization problem, closely related to those in kriging [22], DWO [7] and other similar methods [23]:

$$\mu_k^* = \arg\min_{\mu_k} \quad \mu_k^\top H_1 \mu_k \tag{2a}$$
$$\text{s.t.} \quad \left[\begin{array}{c} D \\ \mathbf{1} \\ U \end{array} \right] \mu_k = \left[\begin{array}{c} z_k \\ 1 \\ u_k \end{array} \right], \tag{2b}$$

where $H_1 \in \mathcal{R}^{N \times N}$ is a positive definite weighting matrix and $\mathbf{1}$ a row vector whose elements are equal to 1. Thanks to the constraints in the optimization problem (2), the decision variable μ_k serves to parameterize an affine hull that contains the current z_k and u_k. The matrix of parameters H_1 is usually chosen as the identity matrix of order N denoted as I_N although other possibilities exist, like weighting the data according to the distance to the current regressor [9].

Note that we are not interested in computing predictions directly, but in obtaining a model of the system parameterized in the weights μ_k. Thus we will introduce some changes in the aforementioned prediction scheme. These changes are related to the computation of the new optimal weights denoted as λ_k^* in such a way that they can be used as the state vector of the proposed model. This implies the need of a state equation in which the successor of λ_k^*, i.e. λ_{k+1}^*, is computed from λ_k^* and u_k. That is, we strive to obtain a state equation of the form $\lambda_{k+1}^* = A\lambda_k^* + Bu_k$. Thus, λ_{k+1}^* will be computed from an optimization problem similar to (2), but ensuring that its solution can be written as the state equation of the model. In order to do so, we first assume that the weights satisfy

$$z_k = D^+ \lambda_k^*, \tag{3a}$$
$$1 = \mathbf{1}\lambda_k^*. \tag{3b}$$

Furthermore, we impose that λ_{k+1}^* must be able to compute z_k and u_k as an affine combination of D and U. Thus, it must satisfy the constraint

$$\left[\begin{array}{c} D \\ \mathbf{1} \\ U \end{array} \right] \lambda_{k+1}^* = \left[\begin{array}{c} z_k \\ 1 \\ u_k \end{array} \right].$$

which is similar to that of problem (2) but aimed to obtain the successor of λ_{k+1}^*, as a function of λ_k^* and u_k. We also add a regularization term penalizing the difference between λ_{k+1}^* and λ_k^* while considering the previous constraint. This

regularization term serves to reduce the sensitivity to noise of the obtained model. Thus, the proposed optimization problem that will lead to the state equation is

$$\begin{aligned}\lambda_{k+1}^* = \arg\min_{\lambda_{k+1}} \; & (\lambda_{k+1}-\lambda_k^*)^\top H_2(\lambda_{k+1}-\lambda_k^*) \\ & + \lambda_{k+1}^\top H_1 \lambda_{k+1} \\ \text{s.t.} \quad & \begin{bmatrix} D \\ \mathbf{1} \\ U \end{bmatrix} \lambda_{k+1} = \begin{bmatrix} z_k \\ 1 \\ u_k \end{bmatrix}.\end{aligned}$$

Note that λ_k^* is assumed to be known from the previous sampling time. Taking into account (3), it is possible to pose the optimization problem as

$$\begin{aligned}\lambda_{k+1}^* = \arg\min_{\lambda_{k+1}} \; & (\lambda_{k+1}-\lambda_k^*)^\top H_2(\lambda_{k+1}-\lambda_k^*) && \text{(4a)} \\ & + \lambda_{k+1}^\top H_1 \lambda_{k+1} \\ \text{s.t.} \quad & C\lambda_{k+1} = C^+\lambda_k^* + \begin{bmatrix} 0 \\ 0 \\ I_{n_u} \end{bmatrix} u_k, && \text{(4b)}\end{aligned}$$

with

$$C = \begin{bmatrix} D \\ \mathbf{1} \\ U \end{bmatrix},\; C^+ = \begin{bmatrix} D^+ \\ \mathbf{1} \\ \mathbf{0} \end{bmatrix},\; H_1 = \tau_1 I_N,\; H_2 = \tau_2 I_N\,,$$

where $\tau_1 > 0$ and $\tau_2 > 0$ are adjustable parameters. Notice that u_k is the current value of the system input, not to be confused with $\bar{u}_i,\ \forall i = 1,\ldots,N$ which are past values of u_k stored in the data base. This problem can be written in canonical form as

$$\begin{aligned}\lambda_{k+1}^* = \arg\min_{\lambda_{k+1}} \; & \frac{1}{2}\lambda_{k+1}^\top H \lambda_{k+1} + f^\top \lambda_{k+1} && \text{(5a)} \\ \text{s.t.} \; & C\lambda_{k+1} = b\,, && \text{(5b)}\end{aligned}$$

with $H = 2(H_1+H_2)$, $f = -2H_2\lambda_k^*$ and $b = C^+\lambda_k^* + \begin{bmatrix} 0 & 0 & I_{n_u} \end{bmatrix}^\top u_k$. Note that we discard the constant term ${\lambda_k^*}^\top H_2 \lambda_k^*$ because we are only interested in the values of λ_{k+1}^*. By means of the Karush-Kuhn-Tucker (KKT) conditions [24, §10.1.1], this problem has the following solution

$$\lambda_{k+1}^* = A\lambda_k^* + Bu_k,$$

with

$$\begin{aligned}A &= 2H^{-1}H_2 + H^{-1}C^\top(CH^{-1}C^\top)^{-1}\left(C^+ - 2CH^{-1}H_2\right), \\ B &= H^{-1}C^\top(CH^{-1}C^\top)^{-1}\begin{bmatrix} 0 & 0 & I_{n_u} \end{bmatrix}^\top.\end{aligned}$$

Then, the state equation of the model is obtained. This would be completed with the output equation of the model which comes from (3). Note that, as we only need the first term of z_k (i.e. the term corresponding to y_k), then we can pose the model as

$$\begin{aligned}\lambda_{k+1}^* &= A\lambda_k^* + Bu_k, && \text{(6a)} \\ y_k &= Y\lambda_k^*\,, && \text{(6b)}\end{aligned}$$

where Y denotes a matrix compounded of only the first n_y rows of D^+, that is, Y is a matrix containing the samples $\bar{y}_i^+$. Thus, we have obtained a model for system (1) in which the state is comprised of the optimal weights λ_k^*.

Remark 1: In order to obtain the initial value λ_0^*, we consider the following ad-hoc optimization problem

$$\begin{aligned}\lambda_0^* = \arg\min_{\lambda_0} \; & \lambda_0^\top H_1 \lambda_0 && \text{(7a)} \\ \text{s.t.} \; & \begin{bmatrix} D^+ \\ \mathbf{1} \end{bmatrix} \lambda_0 = \begin{bmatrix} z_0 \\ 1 \end{bmatrix}, && \text{(7b)}\end{aligned}$$

where z_0 is the initial value of the time delay embedding. This way of computing the initial weights makes sure that equation (3) is fulfilled for every $k \geq 0$. Again, by means of the KKT conditions, the solution of the previous optimization problem is

$$\lambda_0^* = H_1^{-1}\begin{bmatrix} D^+ \\ \mathbf{1} \end{bmatrix}^\top \left(\begin{bmatrix} D^+ \\ \mathbf{1} \end{bmatrix} H_1^{-1} \begin{bmatrix} D^+ \\ \mathbf{1} \end{bmatrix}^\top \right)^{-1} \begin{bmatrix} z_0 \\ 1 \end{bmatrix}.$$

III. Kernel-based State-Space Kriging (K-SSK)

Model (6) is linear on the weights λ_k^*. In order to describe nonlinear behaviors, one could resort to the local data approach of [9]. This approach results in practice on a time-varying linearization of the nonlinear modes of the system (1). Here, we consider the use of kernels to truly model the nonlinear dynamics of the system. In addition to providing a true nonlinear model, the use of kernels will allow us to compute the matrices of the system only once (unlike in [9] where the matrices needed to be recomputed at each sample time). This is very important when the model is to be used in MPC applications, where a recomputation of the system matrices for every candidate solution in the optimization problem would be prohibitive in terms of computational cost. Furthermore, the proposed strategy will allow the formulation of the MPC optimization problem as a quadratic programming problem.

In order to include the kernels, we modify the previous problem to

$$\begin{aligned}\lambda_{k+1}^* = \arg\min_{\lambda_{k+1}} \; & (\lambda_{k+1}-\lambda_k^*)^\top H_2(\lambda_{k+1}-\lambda_k^*) \\ & + \lambda_{k+1}^\top H_1 \lambda_{k+1} \\ & + \left\| \sum_{i=1}^{N} \varphi_{z_i}\lambda_{k+1,i} - \sum_{i=1}^{N} \varphi_{z_i^+}\lambda_{k,i}^* \right\|_{\Sigma_\varphi^{-1}}^2 \\ \text{s.t.} \; & \begin{bmatrix} U \\ \mathbf{1} \end{bmatrix} \lambda_{k+1} = \begin{bmatrix} u_k \\ 1 \end{bmatrix},\end{aligned}$$

where $\lambda_{k+1,i}$ and $\lambda_{k,i}^*$ correspond to the $i-$th element of vector λ_{k+1} and λ_k^* respectively, $\varphi : \mathcal{R}^{n_z} \to \mathcal{H}$ refers to a nonlinear operator that maps $\mathcal{R}^{n_z}$ into a probably high dimensional space $\mathcal{H}$, φ_{z_i} and $\varphi_{z_i^+}$ denote $\varphi(z_i)$ and $\varphi(z_i^+)$ respectively and Σ_φ is a positive definite matrix of appropriate dimensions. In this case, we only need to know the kernel function in order to solve the aforementioned optimization problem [20]. For a given pair $a \in \mathcal{R}^{n_z}$ and $b \in \mathcal{R}^{n_z}$, we denote $\langle \varphi_a, \varphi_b \rangle$ as

$$\langle \varphi_a, \varphi_b \rangle = \varphi_a \Sigma_\varphi^{-1} \varphi_b\,.$$

Note that the previous linear hard constraint on z_k (see equation (4b)) has been changed to a penalty term on a high dimensional space by means of the kernel trick as it is no longer a linear constraint. Assuming that we have data sets

of the evaluation of φ over the time delay embeddings of D and D^+ (which due to the kernel trick [21] are not really necessary), we could denote them as

$$\varphi_{\bar{z}} = \left[\begin{array}{cccc} \varphi_{\bar{z}_1} & \varphi_{\bar{z}_2} & \cdots & \varphi_{\bar{z}_N} \end{array} \right],$$
$$\varphi_{\bar{z}^+} = \left[\begin{array}{cccc} \varphi_{\bar{z}_1^+} & \varphi_{\bar{z}_2^+} & \cdots & \varphi_{\bar{z}_N^+} \end{array} \right].$$

Thus, the previous problem can be written in matrix form as

$$\begin{aligned} \lambda_{k+1}^* = \arg\min_{\lambda_{k+1}} \; & (\lambda_{k+1} - \lambda_k^*)^\top H_2 (\lambda_{k+1} - \lambda_k^*) \\ & + \lambda_{k+1}^\top H_1 \lambda_{k+1} \\ & + \|\varphi_{\bar{z}} \lambda_{k+1} - \varphi_{\bar{z}^+} \lambda_k^*\|^2_{\Sigma_\varphi^{-1}} \\ \text{s.t.} \; & \left[\begin{array}{c} U \\ \mathbf{1} \end{array} \right] \lambda_{k+1} = \left[\begin{array}{c} u_k \\ 1 \end{array} \right]. \end{aligned}$$

Now, we operate with the term $\|\varphi_{\bar{z}} \lambda_{k+1} - \varphi_{\bar{z}^+} \lambda_k^*\|^2_{\Sigma_\varphi^{-1}}$, obtaining

$$\begin{aligned} \|\varphi_{\bar{z}} \lambda_{k+1} - \varphi_{\bar{z}^+} \lambda_k^*\|^2_{\Sigma_\varphi^{-1}} = & \lambda_{k+1}^\top \left(\varphi_{\bar{z}}^\top \Sigma_\varphi^{-1} \varphi_{\bar{z}} \right) \lambda_{k+1} \\ & - 2 \lambda_{k+1}^\top \left(\varphi_{\bar{z}}^\top \Sigma_\varphi^{-1} \varphi_{\bar{z}^+} \right) \lambda_k^* \\ & + \lambda_k^{*\top} \left(\varphi_{\bar{z}^+}^\top \Sigma_\varphi^{-1} \varphi_{\bar{z}^+} \right) \lambda_k^*. \end{aligned}$$

Again, we can discard the constant term because it does not affect the values of λ_{k+1}^*, which leads to the following optimization problem

$$\lambda_{k+1}^* = \arg\min_{\lambda_{k+1}} \; \frac{1}{2} \lambda_{k+1}^\top H \lambda_{k+1} + f^\top \lambda_{k+1} \tag{8a}$$
$$\text{s.t.} \; T \lambda_{k+1} = \left[\begin{array}{c} 0 \\ 1 \end{array} \right] + \left[\begin{array}{c} I_{n_u} \\ 0 \end{array} \right] u_k \,. \tag{8b}$$

with $H = 2(H_1 + H_2) + 2\varphi_{\bar{z}}^\top \Sigma_\varphi^{-1} \varphi_{\bar{z}}$, $f = -2(H_2 + \varphi_{\bar{z}}^\top \Sigma_\varphi^{-1} \varphi_{\bar{z}^+}) \lambda_k^*$, $T = \left[\begin{array}{c} U \\ \mathbf{1} \end{array} \right]$.

Note that the kernel-related terms can be computed because only cross-products appear in the aforementioned equations. These terms would be computed as

$$\varphi_{\bar{z}}^\top \Sigma_\varphi^{-1} \varphi_{\bar{z}} = \left[\begin{array}{cccc} \langle \varphi_{\bar{z}_1}, \varphi_{\bar{z}_1} \rangle & \langle \varphi_{\bar{z}_1}, \varphi_{\bar{z}_2} \rangle & \cdots & \langle \varphi_{\bar{z}_1}, \varphi_{\bar{z}_N} \rangle \\ \langle \varphi_{\bar{z}_2}, \varphi_{\bar{z}_1} \rangle & \langle \varphi_{\bar{z}_2}, \varphi_{\bar{z}_2} \rangle & \cdots & \langle \varphi_{\bar{z}_2}, \varphi_{\bar{z}_N} \rangle \\ \vdots & \vdots & & \vdots \\ \langle \varphi_{\bar{z}_N}, \varphi_{\bar{z}_1} \rangle & \langle \varphi_{\bar{z}_N}, \varphi_{\bar{z}_2} \rangle & \cdots & \langle \varphi_{\bar{z}_N}, \varphi_{\bar{z}_N} \rangle \end{array} \right],$$

$$\varphi_{\bar{z}}^\top \Sigma_\varphi^{-1} \varphi_{\bar{z}^+} = \left[\begin{array}{cccc} \langle \varphi_{\bar{z}_1}, \varphi_{\bar{z}_1^+} \rangle & \langle \varphi_{\bar{z}_1}, \varphi_{\bar{z}_2^+} \rangle & \cdots & \langle \varphi_{\bar{z}_1}, \varphi_{\bar{z}_N^+} \rangle \\ \langle \varphi_{\bar{z}_2}, \varphi_{\bar{z}_1^+} \rangle & \langle \varphi_{\bar{z}_2}, \varphi_{\bar{z}_2^+} \rangle & \cdots & \langle \varphi_{\bar{z}_2}, \varphi_{\bar{z}_N^+} \rangle \\ \vdots & \vdots & & \vdots \\ \langle \varphi_{\bar{z}_N}, \varphi_{\bar{z}_1^+} \rangle & \langle \varphi_{\bar{z}_N}, \varphi_{\bar{z}_2^+} \rangle & \cdots & \langle \varphi_{\bar{z}_N}, \varphi_{\bar{z}_N^+} \rangle \end{array} \right],$$

where $\langle \varphi_{\bar{z}_i}, \varphi_{\bar{z}_j} \rangle$ is the result of applying a certain kernel function with samples $\bar{z}_i$ and $\bar{z}_j$ as inputs. Applying the KKT conditions, the solution of the optimization problem (8) is obtained as

$$\lambda_{k+1}^* = A \lambda_k^* + B u_k + c,$$

with

$$\begin{aligned} A &= 2H^{-1}(I_N - T^\top (T H^{-1} T^\top)^{-1} T H^{-1})(H_2 + \varphi_{\bar{z}}^\top \Sigma_\varphi^{-1} \varphi_{\bar{z}^+}), \\ B &= H^{-1} T^\top (T H^{-1} T^\top)^{-1} \left[\begin{array}{c} I_{n_u} \\ 0 \end{array} \right], \\ c &= H^{-1} T^\top (T H^{-1} T^\top)^{-1} \left[\begin{array}{c} 0 \\ 1 \end{array} \right]. \end{aligned}$$

This equation along with the output equation (6b) provides the new model of the system, i.e.

$$\lambda_{k+1}^* = A \lambda_k^* + B u_k + c \tag{9a}$$
$$y_k = Y \lambda_k^* \,. \tag{9b}$$

Remark 2: Note that this model is affine in the feature space. Assuming that the kernel functions are not linear, the model is nonlinear in the data space. Actually, the model is not strictly linear in the feature space because of the bias term c.

A. Initial condition

As in section II, we need an initial value λ_0^* which can be obtained from

$$\begin{aligned} \lambda_0^* = \arg\min_{\lambda_0} \; & \lambda_0^\top H_1 \lambda_0 + \|\varphi_{z_0} - \varphi_{\bar{z}^+} \lambda_0\|^2_{\Sigma_\varphi^{-1}} \\ \text{s.t.} \; & \mathbf{1} \lambda_0 = 1 \,. \end{aligned}$$

Operating with $\|\varphi_{z_0} - \varphi_{\bar{z}^+} \lambda_0\|^2_{\Sigma_\varphi^{-1}}$, we obtain

$$\lambda_0^* = \arg\min_{\lambda_0} \; \frac{1}{2} \lambda_0^\top H \lambda_0 + f^\top \lambda_0 \tag{10a}$$
$$\text{s.t.} \; \mathbf{1} \lambda_0 = 1 \,, \tag{10b}$$

where

$$H = 2H_1 + 2\varphi_{\bar{z}^+}^\top \Sigma_\varphi^{-1} \varphi_{\bar{z}^+}, \quad f = -2\varphi_{\bar{z}^+}^\top \Sigma_\varphi^{-1} \varphi_{z_0},$$

$$\varphi_{\bar{z}^+}^\top \Sigma_\varphi^{-1} \varphi_{\bar{z}^+} = \left[\begin{array}{cccc} \langle \varphi_{\bar{z}_1^+}, \varphi_{\bar{z}_1^+} \rangle & \langle \varphi_{\bar{z}_1^+}, \varphi_{\bar{z}_2^+} \rangle & \cdots & \langle \varphi_{\bar{z}_1^+}, \varphi_{\bar{z}_N^+} \rangle \\ \langle \varphi_{\bar{z}_2^+}, \varphi_{\bar{z}_1^+} \rangle & \langle \varphi_{\bar{z}_2^+}, \varphi_{\bar{z}_2^+} \rangle & \cdots & \langle \varphi_{\bar{z}_2^+}, \varphi_{\bar{z}_N^+} \rangle \\ \vdots & \vdots & & \vdots \\ \langle \varphi_{\bar{z}_N^+}, \varphi_{\bar{z}_1^+} \rangle & \langle \varphi_{\bar{z}_N^+}, \varphi_{\bar{z}_2^+} \rangle & \cdots & \langle \varphi_{\bar{z}_N^+}, \varphi_{\bar{z}_N^+} \rangle \end{array} \right],$$

$$\varphi_{\bar{z}^+}^\top \Sigma_\varphi^{-1} \varphi_{z_0} = \left[\begin{array}{c} \langle \varphi_{\bar{z}_1^+}, \varphi_{z_0} \rangle \\ \langle \varphi_{\bar{z}_2^+}, \varphi_{z_0} \rangle \\ \vdots \\ \langle \varphi_{\bar{z}_N^+}, \varphi_{z_0} \rangle \end{array} \right].$$

IV. Kalman Filter

Assuming that the obtained model is perfect and there is no noise or disturbances, it holds that

$$\tilde{y}_{k+1} = Y (A \lambda_k^* + B u_k + c) = y_{k+1},$$

where $\tilde{y}_{k+1}$ is the prediction of y_{k+1}. On the other hand, in the presence of noise, this does not hold (i.e. $\tilde{y}_{k+1} \neq y_{k+1}$) although the dynamics of λ_k^* would be correct. For the case where we consider disturbances in the dynamics of λ_k^* (which might represent, for example, model mismatches) it is clear that we cannot compute the true value of λ_k^*. Instead, we can compute a prediction denoted as $\tilde{\lambda}_k^*$. Computing this prediction just by using the proposed update rule corresponds to a form of open-loop prediction because we are not introducing any new information into the analysis. Thus, we consider using the measured outputs to improve the quality of our predicted $\tilde{\lambda}_k^*$, obtaining a corrected version, denoted as $\hat{\lambda}_k^*$. This works as a form of feedback in the computation of the dynamics of the state. In order to implement this strategy, we resort to a Kalman filter as discussed in the following.

For this purpose, we modify the previously obtained nominal model, including disturbances and noisy terms

$$\begin{aligned}\lambda^*_{k+1} =& A\lambda^*_k + Bu_k + c + w_k\\ y_k =& Y\lambda^*_k + v_k\,.\end{aligned}$$

As it is necessary to enforce the equality constraint $\mathbf{1}\lambda^*_k = 1$, the output vector is extended as

$$y'_k = \begin{bmatrix} Y \\ \mathbf{1} \end{bmatrix} \lambda^*_k + \begin{bmatrix} v_k \\ 0 \end{bmatrix}. \tag{11}$$

Defining $G = \begin{bmatrix} Y \\ \mathbf{1} \end{bmatrix}$ and $v'_k = \begin{bmatrix} v_k \\ 0 \end{bmatrix}$, the following extended system is obtained

$$\begin{aligned}\lambda^*_{k+1} &= A\lambda^*_k + Bu_k + c + w_k\\ y'_k &= G\lambda^*_k + v'_k\,.\end{aligned}$$

Due to noise, λ^*_k is also noisy and, thus, it would be helpful to use the optimal prediction of its value obtained by means of a Kalman filter. The prediction of λ^*_k will be denoted as $\tilde{\lambda}^*_k$ whereas the corrected prediction will be denoted as $\hat{\lambda}^*_k$. We assume that w_k and v_k are uncorrelated white noise signals with bounded covariance matrices ς_w and ς_v so that

$$\mathrm{E}(w_k w_k^\top) \leq \varsigma_w, \quad \mathrm{E}(v_k v_k^\top) \leq \varsigma_v,$$

where $\mathrm{E}(\cdot)$ denotes the mathematical expectation. From the bound on the covariance of v_k, it is easy to obtain a bound of v'_k:

$$\mathrm{E}({v'_k}{v'_k}^\top) \leq \mathrm{E}\left(\begin{bmatrix} v_k v_k^\top & 0 \\ 0 & 0 \end{bmatrix}\right) = \begin{bmatrix} \varsigma_v & 0 \\ 0 & 0 \end{bmatrix} = \varsigma'_v\,.$$

The bound of the covariance of the estimation error $\lambda^*_k - \tilde{\lambda}^*_k$ is denoted as $\tilde{\sigma}_k$:

$$\mathrm{E}\left((\lambda^*_k - \tilde{\lambda}^*_k)(\lambda^*_k - \tilde{\lambda}^*_k)^\top\right) \leq \tilde{\sigma}_k.$$

Then, given an estimation $\tilde{\lambda}^*_k$, a corrected version $\hat{\lambda}^*_k$ is obtained from

$$\hat{\lambda}^*_k = \tilde{\lambda}^*_k + S_k\left(\begin{bmatrix} y_k \\ 1 \end{bmatrix} - G\tilde{\lambda}^*_k\right),$$

where S_k is the optimal gain, calculated as

$$S_k = \tilde{\sigma}_k G^\top \left(G\tilde{\sigma}_k G^\top + \varsigma'_v\right)^{-1}.$$

Note that the corrected vector $\hat{\lambda}^*_k$ fulfills the constraint $1 = \mathbf{1}\hat{\lambda}^*_k$ because the last component of the extended output is taken equal to one.

Applying the equations of the Kalman filter, the matrix $\tilde{\sigma}_{k+1}$ is computed as

$$\tilde{\sigma}_{k+1} = A\left(\tilde{\sigma}_k - S_k G\tilde{\sigma}_k\right)A^\top + \varsigma_w\,.$$

Finally, $\tilde{\lambda}^*_{k+1}$ is computed as

$$\tilde{\lambda}^*_{k+1} = A\hat{\lambda}^*_k + Bu_k + c\,.$$

The matrices $\tilde{\sigma}_0, \varsigma_w, \varsigma_v$ are set as the identity matrix multiplied by some scalars that are considered tuning parameters.

V. Illustrative example

Once the K-SSK has been presented, we would like to showcase the advantages of the proposed approach with respect to the Local-Data State-Space Kriging from [9] (LD-SSK). For this purpose, a 1-step ahead forecasting example of the Rössler attractor is presented. Note that we choose an autonomous system so that we can compare with the approach presented in [9]. This system is described by the following set of differential equations

$$\begin{aligned}\dot{o} &= -p - l\\ \dot{p} &= o + ap\\ \dot{l} &= b + l(o - c)\,,\end{aligned}$$

In this example, we consider the set of parameters $a = 0.2$, $b = 0.2$ and $c = 5.7$ which corresponds to chaotic behaviour. Also, it will be considered that the measurements are noisy. This noise follows a uncorrelated normal distribution with zero mean and unit variance $\mathcal{N} \sim (0, 1)$ (that is, the covariance matrix is diagonal).

The sample time of the system is chosen as 0.1 seconds and the data set is compounded of a total of 1000 samples. The regressor considered here is $z_k = [o_k, p_k, l_k]^\top$.

The LD-SSK scheme which incorporates the Kalman filter is used here with parameters $H_2 = 9.79\,I_N$, $\varsigma_w = 1.01 \cdot 10^{-5}\,I_N$ and $\tilde{\sigma}_0 = 6.51 \cdot 10^{-6}\,I_N$. Also, the variance of v_k is assumed to be known.

On the other hand, K-SSK uses a kernel such that

$$\langle \varphi_{\bar{z}_i}, \varphi_{\bar{z}_j} \rangle = \begin{cases} e^{\frac{-||\bar{z}_i - \bar{z}_j||}{2\sigma^2}} + \mathrm{E}(v_k^2) & \text{if } i = j\,,\\ e^{\frac{-||\bar{z}_i - \bar{z}_j||}{2\sigma^2}} & \text{else,}\end{cases}$$

where $\sigma = 0.119$. The value of the matrices H_1 and H_2 are $H_1 = 1.25 \cdot 10^{-5}\,I_N$, $H_2 = 2.24 \cdot 10^{-5}\,I_N$ whereas the parameters of the Kalman filter are $\varsigma_w = 1.23 \cdot 10^{-7}\,I_N$ and $\tilde{\sigma}_0 = 4.09 \cdot 10^{-6}\,I_N$

The results are shown in figure 1 and table I. In figure 1, the real evolution of the state alongside the noisy measurements and the prediction obtained with the K-SSK approach for a representative trajectory is shown. Table I summarizes the numerical results of the experiment. It can be seen that the K-SSK strategy achieves the best results with regard to MSE, standard deviations and computational times.

	LD-SSK	K-SSK
Mean Squared Error	0.2504	0.2138
Standard deviation	0.5002	0.4622
Computational time (ms)	62.909	37.5912

TABLE I
NUMERICAL RESULTS OF THE FORECASTING EXAMPLE.

VI. Tracking Model Predictive Control

A tracking MPC controller [18], [19] that uses the kernel-based SSK model is presented in this section. The main difference of the tracking formulation with respect to the traditional MPC is that the reference to be tracked is considered as an additional decision variable in the optimization problem, that

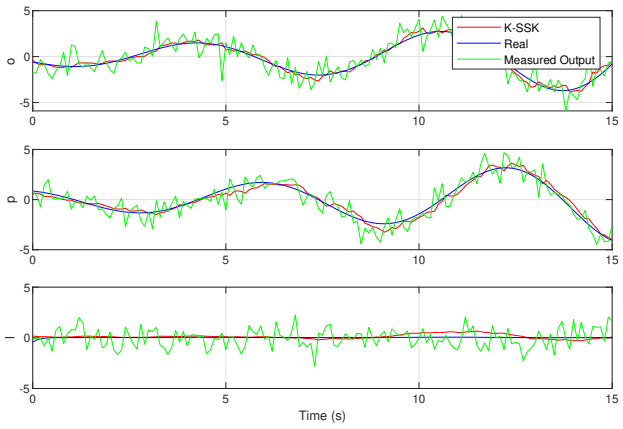

Fig. 1. Forecasting the noisy Rössler attractor with K-SSK.

is, the so-called artificial reference. In order to enforce that the artificial reference converges to the target reference, an additional cost is added to the MPC cost function penalizing the deviation between them. Among the many advantages of this formulation, the fact that recursive feasibility is guaranteed for any change of the desired reference and a significantly larger domain of attraction for short prediction horizons are probably the most important.

The cost function in the proposed controller is comprised of three components:

- A stage cost function $l_s(\cdot,\cdot)$ to penalize tracking error. Tracking error with respect to a certain reference is tackled employing a change of variables $\breve{y} = \tilde{y} - y_s$, $\breve{u} = u - u_s$ where $\tilde{y}$ is a prediction of y, y_s is the artificial output reference and u_s is the artificial input reference. Here, we consider a quadratic cost penalizing the distance to the artificial input and output reference weighted by matrices of appropriate dimensions Q and R (i.e. $Q \in \mathcal{R}^{n_y \times n_y}$ and $R \in \mathcal{R}^{n_u \times n_u}$), i.e.

$$l_s(\breve{y}, \breve{u}) = \breve{y}^\top Q \breve{y} + \breve{u}^\top R \breve{u}.$$

- An offset function penalizing the difference between the artificial output y_s and the target desired reference,

$$l_o(y_s - r) = (y_s - r)^\top O\,(y_s - r)\,.$$

  Under some mild conditions [18], [19], it is proven that the artificial reference converges to the true reference as the time goes by.
- A weighted terminal cost function. This function measures the closeness of the terminal state $\lambda^*_{N_p}$ to the artificial steady state λ^*_s. Appropriately weighting the terminal cost allows us to omit the terminal equality constraint [19], simplifying the design of the controller.

$$l_t(\lambda^*_{N_p} - \lambda^*_s) = \gamma \left(\lambda^*_{N_p} - \lambda^*_s\right)^\top P \left(\lambda^*_{N_p} - \lambda^*_s\right),$$

  where $\gamma \geq 1$.

Denoting $\mathbf{y} \in \mathcal{R}^{n_y N_p}$ and $\mathbf{u} \in \mathcal{R}^{n_u (N_p+1)}$ as $[\tilde{y}_0^\top, \ldots, \tilde{y}_{N_p-1}^\top]^\top$ and $[u_0^\top, \ldots, u_{N_p-1}^\top, u_s^\top]^\top$ respectively, it is possible to define a total cost function $V_{N_p}(\mathbf{y}, \mathbf{u}, r, \lambda^*_0)$ as the sum of the aforementioned three functions for a finite prediction horizon N_p

$$V_{N_p}(\mathbf{y}, \mathbf{u}, r, \lambda^*_0) = \sum_{i=0}^{N_p-1} l_s(\breve{y}_i, \breve{u}_i) + l_o(y_s(u_s) - r) + l_t(\lambda^*_{N_p}(\lambda^*_0, \mathbf{u}) - \lambda_s(u_s)).$$

Thus, we define an MPC control problem with inequality constraints in the outputs and box constraints in the input, that is

$$\min_{\mathbf{y},\mathbf{u},\lambda^*_{N_p}} \quad V_{N_p}(\mathbf{y}, \mathbf{u}, r, \lambda^*_0) \tag{12a}$$

$$\text{s.t.} \quad \lambda^*_{i+1} = A\lambda^*_i + Bu_i + c \quad \forall i = 0, \ldots, N_p - 1 \tag{12b}$$

$$y_i = Y\lambda^*_i \quad \forall i = 0, \ldots, N_p \tag{12c}$$

$$\lambda^*_s = A\lambda^*_s + Bu_s + c \tag{12d}$$

$$\psi y_i \leq \theta \quad \forall i = 1, \ldots, N_p - 1 \tag{12e}$$

$$u_{\min} \leq u_i \leq u_{\max}, \quad \forall i = 0, \ldots, N_p - 1, \tag{12f}$$

which is a parametric quadratic optimization problem whose parameters are r and λ^*_0. Appendix A shows how to pose this problem in canonical form.

From this optimization problem, a sequence of optimal control actions $\mathbf{u}$ is obtained. However, due to the receding horizon scheme characteristic of any MPC controller, only the first component is applied to the system, computing a whole new sequence at the next time instant.

In order to apply the proposed MPC controller, first, we need to compute the system matrices that model the dynamics of the system and obtain the initial value of the vector of weights. This is shown in algorithm 1. Note that algorithm 1 is only executed once. Once this has been done, the detailed steps to compute the MPC controller by using the Kalman filter layer are detailed in algorithm 2. Unlike algorithm 1, algorithm 2 is executed at each time instant k.

A. Nominal stability analysis

For the nominal stability analysis, it is assumed that there are no mismatches between the dynamics of the real system and the obtained prediction model and that there is no noise in the measurements. Therefore, the state vector λ^*_k is assumed to be perfectly estimated by the Kalman Filter. Now, we present the assumptions appearing in [19] to guarantee the nominal stability of the controller for completeness:

Assumption 6.1: The output of the system y_s univocally defines the equilibrium point (λ^*_s, u_s). Also, it is assumed that λ^*_s and u_s can be obtained by means of locally Lipschitz continuous functions $g_\lambda : \mathcal{R}^{n_y} \to \mathcal{R}^N$ and $g_u : \mathcal{R}^{n_y} \to \mathcal{R}^{n_u}$ so that

$$\lambda^*_s = g_\lambda(y_s), \quad u_s = g_u(y_s).$$

The stage cost function, the offset cost function and the set of feasible points need to satisfy the following assumptions:

Assumption 6.2:

1) There exists a $\mathcal{K}_\infty$ function α_l such that $l_s(y, u) \geq \alpha_l(|y|)$ for every pair $(y, u) \in \mathcal{R}^{n_y + n_u}$.

Algorithm 1 Computation of the system matrices and initial conditions.

Require: Matrices H_1, H_2 and T, kernel function.

Ensure: System matrices A, B and c, initial weights λ_0^*.

1: Compute the kernel matrices:

$$\varphi_{\bar{z}}^\top \Sigma_\varphi^{-1} \varphi_{\bar{z}} = \begin{bmatrix} \langle \varphi_{\bar{z}_1}, \varphi_{\bar{z}_1} \rangle & \langle \varphi_{\bar{z}_1}, \varphi_{\bar{z}_2} \rangle & \cdots & \langle \varphi_{\bar{z}_1}, \varphi_{\bar{z}_N} \rangle \\ \langle \varphi_{\bar{z}_2}, \varphi_{\bar{z}_1} \rangle & \langle \varphi_{\bar{z}_2}, \varphi_{\bar{z}_2} \rangle & \cdots & \langle \varphi_{\bar{z}_2}, \varphi_{\bar{z}_N} \rangle \\ \vdots & \vdots & & \vdots \\ \langle \varphi_{\bar{z}_N}, \varphi_{\bar{z}_1} \rangle & \langle \varphi_{\bar{z}_N}, \varphi_{\bar{z}_2} \rangle & \cdots & \langle \varphi_{\bar{z}_N}, \varphi_{\bar{z}_N} \rangle \end{bmatrix},$$

$$\varphi_{\bar{z}}^\top \Sigma_\varphi^{-1} \varphi_{\bar{z}^+} = \begin{bmatrix} \langle \varphi_{\bar{z}_1}, \varphi_{\bar{z}_1^+} \rangle & \langle \varphi_{\bar{z}_1}, \varphi_{\bar{z}_2^+} \rangle & \cdots & \langle \varphi_{\bar{z}_1}, \varphi_{\bar{z}_N^+} \rangle \\ \langle \varphi_{\bar{z}_2}, \varphi_{\bar{z}_1^+} \rangle & \langle \varphi_{\bar{z}_2}, \varphi_{\bar{z}_2^+} \rangle & \cdots & \langle \varphi_{\bar{z}_2}, \varphi_{\bar{z}_N^+} \rangle \\ \vdots & \vdots & & \vdots \\ \langle \varphi_{\bar{z}_N}, \varphi_{\bar{z}_1^+} \rangle & \langle \varphi_{\bar{z}_N}, \varphi_{\bar{z}_2^+} \rangle & \cdots & \langle \varphi_{\bar{z}_N}, \varphi_{\bar{z}_N^+} \rangle \end{bmatrix},$$

$$\varphi_{\bar{z}^+}^\top \Sigma_\varphi^{-1} \varphi_{\bar{z}^+} = \begin{bmatrix} \langle \varphi_{\bar{z}_1^+}, \varphi_{\bar{z}_1^+} \rangle & \langle \varphi_{\bar{z}_1^+}, \varphi_{\bar{z}_2^+} \rangle & \cdots & \langle \varphi_{\bar{z}_1^+}, \varphi_{\bar{z}_N^+} \rangle \\ \langle \varphi_{\bar{z}_2^+}, \varphi_{\bar{z}_1^+} \rangle & \langle \varphi_{\bar{z}_2^+}, \varphi_{\bar{z}_2^+} \rangle & \cdots & \langle \varphi_{\bar{z}_2^+}, \varphi_{\bar{z}_N^+} \rangle \\ \vdots & \vdots & & \vdots \\ \langle \varphi_{\bar{z}_N^+}, \varphi_{\bar{z}_1^+} \rangle & \langle \varphi_{\bar{z}_N^+}, \varphi_{\bar{z}_2^+} \rangle & \cdots & \langle \varphi_{\bar{z}_N^+}, \varphi_{\bar{z}_N^+} \rangle \end{bmatrix},$$

$$\varphi_{\bar{z}^+}^\top \Sigma_\varphi^{-1} \varphi_{z_0} = \begin{bmatrix} \langle \varphi_{\bar{z}_1^+}, \varphi_{z_0} \rangle \\ \langle \varphi_{\bar{z}_2^+}, \varphi_{z_0} \rangle \\ \vdots \\ \langle \varphi_{\bar{z}_N^+}, \varphi_{z_0} \rangle \end{bmatrix}.$$

2: Let $H = 2(H_1 + H_2) + 2\varphi_{\bar{z}}^\top \Sigma_\varphi^{-1} \varphi_{\bar{z}}$.

3: Compute the system matrices A, B and c:

$$A = 2H^{-1}(I_N - T^\top (TH^{-1}T^\top)^{-1} TH^{-1})(H_2 + \varphi_{\bar{z}}^\top \Sigma_\varphi^{-1} \varphi_{\bar{z}^+}),$$
$$B = H^{-1}T^\top (TH^{-1}T^\top)^{-1} \begin{bmatrix} I_{n_u} \\ 0 \end{bmatrix},$$
$$c = H^{-1}T^\top (TH^{-1}T^\top)^{-1} \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

4: Obtain λ_0^* by solving the following optimization problem:

$$\lambda_0^* = \arg\min_{\lambda_0} \ \frac{1}{2}\lambda_0^\top H \lambda_0 + f^\top \lambda_0$$
$$\text{s.t. } \mathbf{1}\lambda_0 = 1,$$

where

$$H = 2H_1 + 2\varphi_{\bar{z}^+}^\top \Sigma_\varphi^{-1} \varphi_{\bar{z}^+}, \quad f = -2\varphi_{\bar{z}^+}^\top \Sigma_\varphi^{-1} \varphi_{z_0}.$$

Algorithm 2 SSK-MPC for tracking controller with Kalman filtering.

Require: System matrices A, B and c, data set Y, desired reference r, weighting matrices Q, R, O and P, prediction horizon N_p, corrected state $\hat{\lambda}_{k-1}$, covariance matrix $\tilde{\sigma}_{k-1}$, covariance matrices ς_w and ς_v'.

Ensure: Optimal input u_k^*, covariance matrix $\tilde{\sigma}_k$ and corrected state $\hat{\lambda}_k$.

1: Compute the prediction of λ_k:

$$\tilde{\lambda}_k^* = A\hat{\lambda}_{k-1}^* + Bu_{k-1} + c\,.$$

2: Compute the matrix $\tilde{\sigma}_k$ as:

$$\tilde{\sigma}_k = A\left(\tilde{\sigma}_{k-1} - S_{k-1}G\tilde{\sigma}_{k-1}\right)A^T + \varsigma_w\,.$$

3: Computation of the optimal gain:

$$S_k = \tilde{\sigma}_k G^T \left(G\tilde{\sigma}_k G^T + \varsigma_v'\right)^{-1}.$$

4: Obtain the corrected version of $\tilde{\lambda}_k$:

$$\hat{\lambda}_k^* = \tilde{\lambda}_k^* + S_k\left(\begin{bmatrix} y_k \\ 1 \end{bmatrix} - G\tilde{\lambda}_k^*\right).$$

5: Solve the quadratic optimization problem:

$$\begin{aligned} \min_{\mathbf{y},\mathbf{u},\lambda_{N_p}^*} \quad & V_{N_p}(\mathbf{y}, \mathbf{u}, r, \hat{\lambda}_k^*) \\ \text{s.t. } \quad & \lambda_{i+1}^* = A\lambda_i^* + Bu_i + c \quad \forall i = 0, \ldots, N_p - 1 \\ & y_i = Y\lambda_i^* \quad \forall i = 0, \ldots, N_p \\ & \lambda_s^* = A\lambda_s^* + Bu_s + c \\ & \psi y_i \le \delta \quad \forall i = 1, \ldots, N_p - 1 \\ & u_{\min} \le u_i \le u_{\max}, \quad \forall i = 0, \ldots, N_p - 1. \end{aligned}$$

6: Choose u_k^* as the first term in the optimal sequence $\mathbf{u}^*$.

2) Denote $\mathcal{Y}_t$ and $\mathcal{Y}_s$ as the set of feasible points and the set of admissible points of the output. Then, $\mathcal{Y}_t$ must be a convex subset of $\mathcal{Y}_s$.

3) The offset cost function $l_o : \mathcal{R}^{n_y} \to \mathcal{R}$ is a subdifferentiable convex positive definite function such that

$$y_s^* = \arg\min_{y_s \in \mathcal{Y}_t} l_o(y_s - r),$$

is unique. Also, there exists a $\mathcal{K}_\infty$ function α_o such that

$$l_o(y_s - r) - l_o(y_s^* - r) \ge \alpha_o(|y_s - y_s^*|).$$

Finally, the terminal ingredients must fulfill the next assumptions:

Assumption 6.3:

1) Let $\kappa(\lambda^*, y_s)$ be a continuous control policy in (λ_s^*, y_s) and Γ be an associated invariant set for tracking such that for any pair $(\lambda_{(0)}^*, y_s) \in \Gamma$ the closed-loop system $\lambda_{k+1}^* = A\lambda_k + B\kappa(\lambda_k^*, y_s) + c$ is asymptotically stable and converges to the equilibrium point (λ_s^*, y_s) in such a way that $(\lambda_k^*, y_s) \in \Gamma$ and the constraints are satisfied for all $k \ge 0$.

2) l_t is a Lyapunov function for the system $\lambda_{k+1}^* = A\lambda_k^* + B\kappa(\lambda_k^*, y_s) + c$ such that for every pair $(\lambda^*, y_s) \in \Gamma$, there exist constants $b > 0$ and $\sigma > 1$ verifying

$$l_t(\lambda^* - \lambda_s^*) \le b|\lambda^* - \lambda_s^*|^\sigma,$$

and

$$l_t(A\lambda^* + B\kappa(\lambda^*, y_s) + c - \lambda_s^*) - l_t(\lambda^* - \lambda_s^*) \le l_s(y - y_s, \kappa(\lambda^*, y_s) - u_s). \quad (13)$$

By using the previous assumptions, the proof of stability for the closed-loop system is a direct application of theorem 3 from [19]. Thus, it is only needed to ensure that the aforementioned assumptions hold.

1) Assumption 6.1 can be easily fulfilled by using remark 1 in [19]. Here, it suffices that matrix

$$\begin{bmatrix} A - I_N & B \\ Y & 0 \end{bmatrix},$$

is nonsingular, which holds true if the number of outputs is equal to the number of inputs.

2) For assumption 6.2, since we have chosen a quadratic stage cost function and the system is linear, we can easily find a $\mathcal{K}_\infty$ function that works as a lower bound. Also, as the offset cost function is convex and quadratic, we have that y_s^* is unique and the value of this cost function can be lower bounded with any arbitrarily small quadratic function as well. Finally, as we are considering an affine system with convex constraints, $\mathcal{Y}_t$ can be chosen as $\mathcal{Y}_s$ because $\mathcal{Y}_s$ is already a convex set.
3) Regarding assumption 6.3, taking into account the fact that the terminal cost function is quadratic, it can be easily upper bounded by using $b > 0$ and $\sigma \geq 2$. On the other hand, we choose the matrix P by solving the Lyapunov equation

$$(A + BK)^\top P(A + BK) - P = -(Q + K^\top RK),$$

where K corresponds to the gain of a linear control law such that

$$\kappa(\lambda^*, y_s) = K(\lambda^* - g_\lambda(y_s)) + g_u(y_s).$$

Then, equation (13) is fulfilled and the system converges to the pair $(g_\lambda(y_s), g_u(y_s))$.

B. Robust stability analysis

In this section, we tackle the case where modeling errors may appear due to the fact that the model of the system is not perfect. Here, we focus on the *Robustly Asymptotically Stability* (RAS) notion showcased in [25]. Denote e_k as measurement errors (i.e. $e_k = \tilde{\lambda}_k^* - \lambda_k^*$) and w_k as additive disturbances. Also, denote $\mathbf{e}$ and $\mathbf{w}$ as sequences of N_p elements of e_k and w_k. Then, the definition of RAS is the following.

Definition 6.1: The system $\lambda_{k+1}^* = A\lambda_k^* + B\kappa(\lambda_k^* + e_k, y_s) + c + w_k$ is considered to be RAS in the interior of a set $\mathcal{F}$ with respect to both measurement errors and additive disturbances if it is possible to find a $\mathcal{KL}$ function β [26] satisfying that, for each $\epsilon > 0$ and compact set $\mathcal{C} \subset \mathcal{F}$, there exists $\delta > 0$ such that, for every pair $(\mathbf{w}, \mathbf{e})$, the following is fulfilled:

1) $\max(\mathbf{w}) \leq \delta$, $\max(\mathbf{e}) \leq \delta$ and,
2) the closed loop trajectory under such $(\mathbf{w}, \mathbf{e})$ belongs to $\mathcal{C}$.

Then, we have that the closed loop trajectory is bounded by $\beta(|x|, k) + \epsilon$.

In light of proposition 8 from [25], we have that if the nominal system is asymptotically stable with an associated continuous Lyapunov function, then the system is RAS. Since nominal stability of the proposed controller has been previously shown, then we only need to prove that the optimal cost function that serves as the Lyapunov function is continuous. This property holds thanks to the fact that the resulting optimization control problem is a multi-parametric QP in (λ, y_s); then its optimal cost is a continuous function of the parameters.

VII. Numerical Examples

This section presents two application examples of the proposed controller. First, the application to a simulated single-input single-output system, a continuously stirred tank reactor, is presented, followed by the application to a temperature control equipment.

A. Simulation case study: Continuously-Stirred Tank Reactor

First, we design an MPC controller for a Continuously-Stirred Tank Reactor (CSTR) [27]. The dynamics of the system are given by the following set of differential equations

$$\begin{aligned}\frac{dC_A(t)}{dt} =& \frac{q_0}{V}(C_{Af} - C_A(t)) - k_0 e^{\frac{-E}{RT(t)}} C_A(t),\\ \frac{dT(t)}{dt} =& \frac{q_0}{V}(T_f - T(t)) + \frac{(-\Delta H_r)k_0}{\rho C_p} e^{\frac{-E}{RT(t)}} C_A(t)\\ &+ \frac{UA}{V\rho C_p}(T_c(t) - T(t)),\end{aligned}$$

whose parameters are shown in table II. The system has a unique input T_c (temperature of the cooling jacket) whereas the output is C_A, the concentration of the reactant. The regressor is compounded of the last two values of the output $z_k = [y_k, y_{k-1}]^\top$. The sampling time is $t_s = 0.5$ min. The data sets D, D^+ and U are obtained from a simulation experiment where random amplitude step input signals are applied to the system. A total of $N = 250$ samples are considered. Also, it is assumed that the measurements are noisy, having an additive gaussian noise v_k whose variance is $2.5 \cdot 10^{-5}$. The controller considers input constraints as

$$u_{\min} \leq u_k \leq u_{\max},\ \forall k = 1, ..., N_p$$

where $u_{\min} = 341.5K$, $u_{\max} = 365.5K$ and $N_p = 10$. The weighting parameters Q, R, O, H_1, H_2 and γ are

$$\begin{aligned}Q = 500, \quad R = 0.02, \quad O = 5000\\ H_1 = 0.002I_N, \quad H_2 = 0.007I_N, \quad \gamma = 10.\end{aligned}$$

On the other hand, the chosen *radial basis kernel* is

$$\langle \varphi_{\bar{z}_i}, \varphi_{\bar{z}_j} \rangle = e^{\frac{-||\bar{z}_i - \bar{z}_j||}{2\sigma^2}}$$

with $\sigma = 7.6741$. Note that the parameters of the SSK approach (i.e. n_p and the parameters of the kernel function) are chosen so that the mean-squared error (MSE) of some open-loop predictions in a validation set compounded of 1750 samples is minimized. The output and input data are scaled as

$$y = \frac{C_A - C_A^{\min}}{C_A^{\max} - C_A^{\min}},\ u = T_c - 353.5\,\mathrm{K}\,.$$

For the Kalman filter, the following parameters are considered

$$\varsigma_w = 5 \times 10^{-6} I_N,\ \varsigma_v = 2.5 \cdot 10^{-5},\ \sigma_0 = 0.1 I_N.$$

The results are shown in figure 2. The blue line corresponds to the reference both in the input and the output whereas the red line corresponds to the real output and the inputs applied to the system. It is clear that the controller steers the system to the desired references. However, due to the noisy measurements and the discrepancies of the model with respect

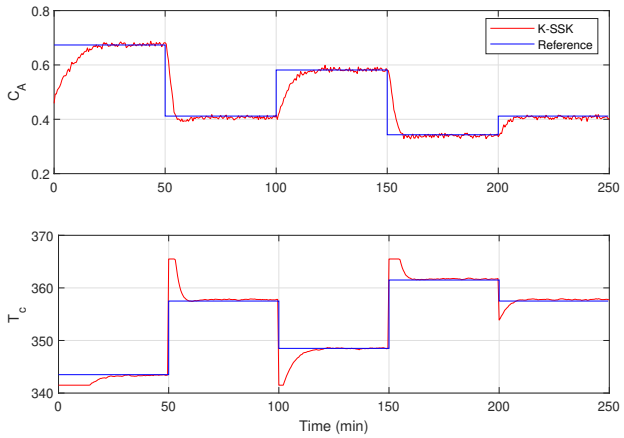

Fig. 2. Top figure: output of the system. Bottom figure: input applied to the system.

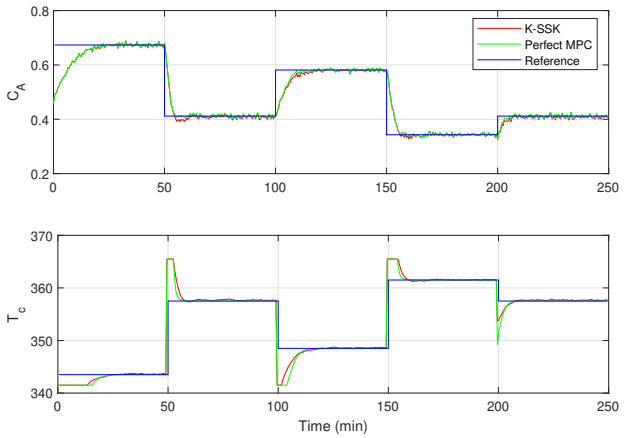

Fig. 3. Top figure: output of the system. Bottom figure: input applied to the system.

to the real system, it is possible to see some offset. This is something to be expected and several strategies could be used to solve this, e.g. using an appropriate disturbance model [28] or augmenting the state [29] (see [30] for a tutorial on the subject).

Parameter	Meaning	Value	Units
q_0	Input flow of the reactive	10	l min^{-1}
V	Liquid volume in the tank	150	l
k_0	Frecuency constant	6×10^{10}	min^{-1}
E/R	Arrhenius constant	9750	K
$-\Delta H_r$	Enthalpy of the reaction	10000	J mol^{-1}
UA	Heat transfer coefficient	70000	J $\text{min}^{-1}\text{K}^{-1}$
ρ	Density	1100	g l^{-1}
C_p	Specific heat	0.3	J $\text{g}^{-1}\text{K}^{-1}$
C_{Af}	C_A in the input flow	1	mol l^{-1}
T_f	Temperature (input flow)	370	K

TABLE II
PARAMETERS OF THE CSTR MODEL

1) Comparison with an ideal NMPC controller: We consider an ideal MPC controller for comparison purposes. This means that the model of the system is completely known for this benchmark controller. Also, this controller can measure the whole state vector and thus the variables used to compute the control actions are noiseless. On the other hand, the proposed approach can only obtain noisy measurements of the output of the system and it does not have any previous knowledge about the process dynamics.

The results are shown in figure 3. Again, the red line corresponds to the proposed controller, the blue line corresponds to the desired reference and, now, the green line corresponds to the perfect MPC controller. It is clear that the obtained trajectory is very similar with respect to the *perfect* trajectory. The closed-loop cost for our proposed controller is only 1.0337% larger, which is quite reasonable taking into account that the ideal controller cannot be implemented in practice and the proposed controller has no model to start with. Also, the time needed to solve the optimization problem of the proposed approach was 3.9440 ms on average whereas the ideal controller takes an average time of 10.214 ms.

B. Experimental case study: Temperature Control Lab

The temperature control lab is an application of feedback control made up of an Arduino and a shield with two heaters and two temperature sensors (see figure 4). The heaters correspond to the inputs whereas the temperatures are the outputs of the system. The heater power output can be adjusted so that a certain temperature reference is attained. The thermal energy within the system is transferred by conduction, convection, and radiation. Also, heat is transferred away from the device to the surroundings [31].

The dynamics of the system can be modeled by means of the following two energy balance equations

$$\begin{aligned} m\, c_p \frac{dT_1}{dt} &= U\, A\, (T_\infty - T_1) + \epsilon\, \sigma\, A\, (T_\infty^4 - T_1^4) \\ &\quad + Q_{C12} + Q_{R12} + \alpha_1 u_1, \\ m\, c_p \frac{dT_2}{dt} &= U\, A\, (T_\infty - T_2) + \epsilon\, \sigma\, A\, (T_\infty^4 - T_2^4) \\ &\quad - Q_{C12} - Q_{R12} + \alpha_2 u_2, \end{aligned}$$

where T_1 and T_2 are the temperatures of the sensors, T_∞ is the ambient temperature, u_1 and u_2 are the inputs, which are given in a ratio manner (i.e. $u_i = 0$ means that no power is being injected into the system and $u_i = 1$ that the input is upper saturated, injecting all its possible power), m, c_p, U, A, ϵ, σ, α_1, α_2 are constants whose approximate values can be found in table III, Q_{C12} corresponds to the convective heat transfer and Q_{R12} is the radiation heat transfer. These heat flows are given by

$$\begin{aligned} Q_{C12} &= U\, A_s\, (T_2 - T_1), \\ Q_{R12} &= \epsilon\, \sigma\, A\, (T_2^4 - T_1^4). \end{aligned}$$

The data sets D, D^+ and U are comprised of past trajectories of the real system. These are obtained from an experiment where random step signals are applied to the system as can be seen in figure 5. There are a total of $N = 476$ data points. The regressor is compounded of the last values of the output $z_k = [T_{1,k}, T_{2,k}]^\top$ where T_1 and T_2 correspond to the aforementioned output temperatures. The sampling time

Parameter	Meaning	Value	Units
α_1	Heater factor	0.01	W/(%)
α_2	Heater factor	0.0075	W/(%)
m	Mass	0.004	kg
c_p	Heat capacity	500	J kg^{-1}K^{-1}
A	Area Not Between Heaters	1×10^{-3}	m^2
A_s	Area Between Heaters	2×10^{-4}	m^2
U	Heat Transfer Coefficient	10	W m^{-2}K^{-1}
ϵ	Emissivity	0.9	None
σ	Stefan-Boltzmann Constant	5.67×10^{-8}	W m^{-2}K^{-4}

TABLE III
PARAMETERS OF THE MODEL FOR THE TEMPERATURE CONTROL LAB.

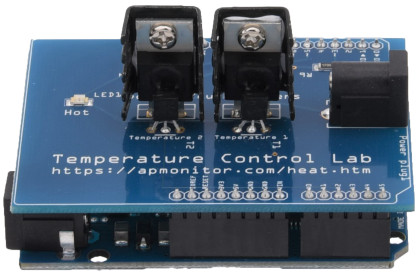

Fig. 4. Temperature Control Lab.

is set to $t_s = 0.5$ min. Same as before, input constraints are considered as

$$u_{\min} \leq u_k \leq u_{\max}, \ \forall k = 1, ..., N_p$$

where $u_{\min} = 0$, $u_{\max} = 100$ and $N_p = 10$. The weighting parameters Q, R, O, H_1, H_2 and γ are

$$Q = 10I_{n_y}, \quad R = 1I_{n_u}, \quad O = 1000I_{n_y}$$
$$H_1 = 0.005I_N, \quad H_2 = 0.008I_N, \quad \gamma = 1.$$

As in the previous example, a *radial basis kernel* is used with $\sigma = 0.413$ and the output and input data is scaled, making the data range from 0 to 1. The parameters of the proposed approach are obtained by minimizing the MSE of the open-loop predictions. Due to the fact that there is a scarce number of samples, the training set is used as a validation set as well. On the other hand, the parameters of the Kalman Filter are the following

$$\varsigma_w = 10^{-6}I_N, \ \varsigma_v = I_{n_y}, \ \sigma_0 = 0.05I_N.$$

The results are shown in figure 6. The red line corresponds to the temperature of the first heater T_1 and the value of the first input u_1, the blue line corresponds to the temperature of the second heater T_2 and the second input u_2 whereas the black-dashed line is the reference for T_1 and u_1 and the green-dashed line is the reference for T_2 and u_2. The MPC controller activates at $t = 5$ min. As in the previous example, the controller is able to steer the outputs of the system to the desired references. Also, there is some minor offset, which is more evident in the control actions plots, where it is evident that their values at equilibrium (y_s, u_s) do not match with those expected from the data set (dotted plot). This is due to the unknown ambient temperature that may vary considerably from one day to another or even within the same day. The experiment of figure 6 and the data set were obtained on

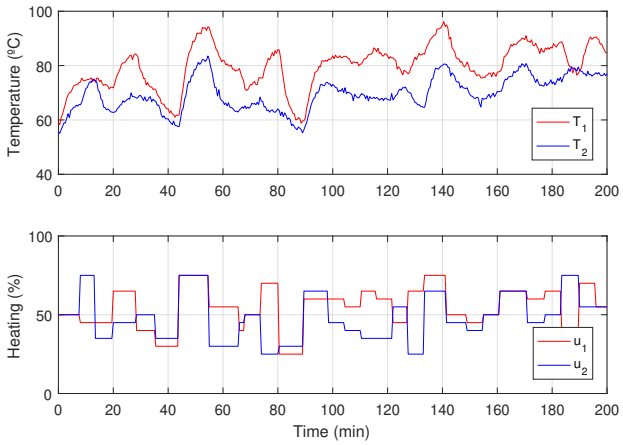

Fig. 5. Experimental data obtained: Top figure: output of the system. Bottom figure: input applied to the system.

different days, thus, the equilibrium points are not exactly congruent.

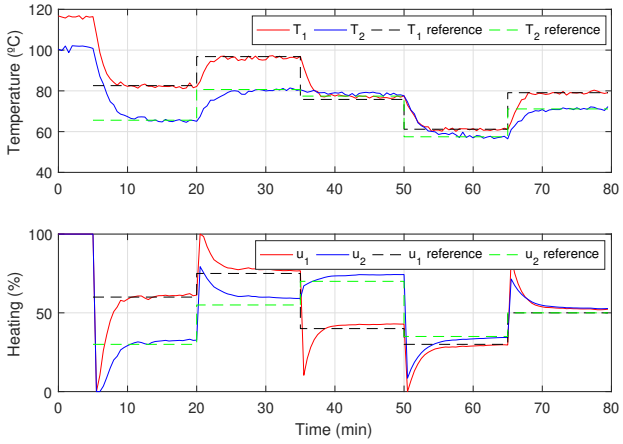

Fig. 6. Top figure: outputs of the system. Bottom figure: inputs applied to the system.

VIII. CONCLUSIONS

A tracking MPC controller using an SSK prediction scheme has been presented in this paper. To be useful for this application, the SSK method has been extended to consider non-autonomous systems. Furthermore, kernel functions are proposed to deal with system nonlinearities. The controller considers hard constraints on the inputs and outputs. The controller has been tested in two different application examples. The first one is a univariate simulated process, a CSTR. On the other hand, the second example shows experimental results with a laboratory temperature control process with two inputs and two outputs. Both examples yielded good results, thus validating the proposed controller and prediction scheme. Future works include both the integration of a real-time optimizer and the consideration of economic objectives in the controller.

APPENDIX A

As shown in the following, problem (12) can be posed as a canonical quadratic programming problem. For that purpose, using the previous definition of the system shown in (9), it is clear that, given a certain λ_0^*, λ_k^* can be obtained by iterating λ_0^* repeatedly forward

$$\lambda_k^* = A^k \lambda_0^* + \sum_{i=0}^{k-1} A^{k-1-i}(Bu_i + c)\,.$$

As $y_k = Y\lambda_k^*$, then

$$\tilde{y}_k = YA^k \lambda_0^* + \sum_{i=0}^{k-1} YA^{k-1-i}(Bu_i + c)\,. \tag{14}$$

Equation (14) can be written in matrix form as

$$\mathbf{y} = \mathbf{n} + \mathbf{M}\mathbf{u}\,, \tag{15}$$

where

$$\mathbf{n} = \begin{bmatrix} Y\lambda_0^* \\ YA\lambda_0^* + c \\ \vdots \\ YA^{N_p-1}\lambda_0^* + \sum_{i=0}^{N_p-2} YA^{N_p-2-i}c \end{bmatrix},$$

$$\mathbf{M} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 0 \\ YB & 0 & 0 & \cdots & 0 & 0 \\ YAB & YB & 0 & \cdots & 0 & 0 \\ YA^2B & YAB & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & & \vdots & \vdots \\ YA^{N_p-2}B & YA^{N_p-3}B & \cdots & YB & 0 & 0 \end{bmatrix}.$$

Then, the constraints (12e) and (12f) can be posed as

$$\Psi\mathbf{M}\mathbf{u} \leq \Theta - \Psi\mathbf{n}$$

where

$$\Psi = \begin{bmatrix} \psi & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \psi \end{bmatrix}, \quad \Theta = \begin{bmatrix} \theta \\ \vdots \\ \theta \end{bmatrix},$$

and

$$\underline{\mathbf{u}} \leq \mathbf{u} \leq \overline{\mathbf{u}}$$

where

$$\underline{\mathbf{u}} = \begin{bmatrix} u_{\min} \\ \vdots \\ u_{\min} \end{bmatrix}, \quad \overline{\mathbf{u}} = \begin{bmatrix} u_{\max} \\ \vdots \\ u_{\max} \end{bmatrix}.$$

On the other hand, as it was previously stated, (λ_s^*, u_s) are an equilibrium pair, thus they satisfy

$$\lambda_s^* = A\lambda_s^* + Bu_s + c,$$

then, they can be obtained as

$$\lambda_s^* = (I_N - A)^{-1}(Bu_s + c)\,,$$
$$y_s = Y(I_N - A)^{-1}(Bu_s + c)\,,$$

which can be written as

$$\lambda_s^* = W\mathbf{u} + v,$$
$$y_s = YW\mathbf{u} + Yv,$$

where

$$W = \begin{bmatrix} 0 & 0 & \ldots & 0 & (I-A)^{-1}B \end{bmatrix},$$
$$v = (I-A)^{-1}c.$$

Also, it is convenient to define

$$\mathbf{y}_s = \begin{bmatrix} y_s \\ \vdots \\ y_s \end{bmatrix} = \mathbf{W}_Y\mathbf{u} + \mathbf{v}_Y,$$

where

$$\mathbf{W}_Y = \begin{bmatrix} YW \\ \vdots \\ YW \end{bmatrix}, \mathbf{v}_Y = \begin{bmatrix} Yv \\ \vdots \\ Yv \end{bmatrix}.$$

Once reached this point, it is possible to write the output tracking error term in matrix form. Now, making

$$\mathbf{L} = \begin{bmatrix} I_{n_u} & 0 & 0 & -I_{n_u} \\ \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & I_{n_u} & -I_{n_u} \end{bmatrix},$$

$$\mathbf{Q} = \begin{bmatrix} Q & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & Q \end{bmatrix}, \mathbf{R} = \begin{bmatrix} R & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & R \end{bmatrix},$$

we obtain

$$\sum_{i=0}^{N_p-1} l_s(\breve{y}_i, \breve{u}_i) = (\mathbf{y} - \mathbf{y}_s)^\top \mathbf{Q}(\mathbf{y} - \mathbf{y}_s) + \mathbf{u}^\top \mathbf{L}^\top \mathbf{R}\mathbf{L}\mathbf{u}$$
$$= (\mathbf{M}\mathbf{u} + \mathbf{n} - \mathbf{W}_Y\mathbf{u} + \mathbf{v}_Y)^\top \mathbf{Q}(\mathbf{M}\mathbf{u} + \mathbf{n} - \mathbf{W}_Y\mathbf{u} + \mathbf{v}_Y)$$
$$+ \mathbf{u}^\top \mathbf{L}^\top \mathbf{R}\mathbf{L}\mathbf{u}.$$

Also, $l_o(y_s - r)$ can be written as

$$l_o(y_s - r) = (YW\mathbf{u} + Yv - r)^\top O(YW\mathbf{u} + Yv - r)\,.$$

It only remains to obtain a matrix expression for $l_t(\lambda_{N_p}^*, \lambda_s^*)$. As it was already shown how to obtain λ_s^*, it is only needed to show how to obtain $\lambda_{N_p}^*$. Using equation (14), we obtain

$$\lambda_{N_p}^* = T\mathbf{u} + h,$$

where

$$T = \begin{bmatrix} A^{N_p-1}B & A^{N_p-2}B & \cdots & B & 0 \end{bmatrix},$$
$$h = A^{N_p-1}\lambda_0^* + \sum_{i=0}^{N_p-1} A^{N_p-1-i}c,$$

and thus

$$l_t(\lambda_{N_p}^* - \lambda_s^*) = \gamma(T\mathbf{u} + h - W\mathbf{u} - v)^\top P(T\mathbf{u} + h - W\mathbf{u} - v)\,.$$

It is easy to see that the optimization problem in (12) becomes

$$\min_{\mathbf{u}} \ \frac{1}{2}\mathbf{u}^\top H\mathbf{u} + f^\top\mathbf{u} + f_0 \tag{16a}$$
$$\text{s.t.} \ \ \Psi\mathbf{M}\mathbf{u} \leq \Theta - \Psi\mathbf{n} \tag{16b}$$
$$\underline{\mathbf{u}} \leq \mathbf{u} \leq \overline{\mathbf{u}}. \tag{16c}$$

where

$$
\begin{aligned}
H &= 2((\mathbf{M}-\mathbf{W}_Y)^\top \mathbf{Q}(\mathbf{M}-\mathbf{W}_Y)+\mathbf{L}^\top \mathbf{R}\mathbf{L} \\
&\quad + W^\top Y^\top O Y W + \gamma((T-W)^\top P(T-W))), \\
f^\top &= 2(\mathbf{n}-\mathbf{v}_Y)^\top \mathbf{Q}(\mathbf{M}-\mathbf{W}_Y) + 2(Yv-r)^\top O Y W \\
&\quad + 2\gamma(h-v)^\top P(T-W), \\
f_0 &= (\mathbf{n}-\mathbf{v}_Y)^\top \mathbf{Q}(\mathbf{n}-\mathbf{v}_Y) + (Yv-r)^\top O(Yv-r) \\
&\quad + \gamma(h-v)^\top P(h-v).
\end{aligned}
$$

Thus, the control action is computed as

$$
\begin{aligned}
\mathbf{u}^* = \arg\min_{\mathbf{u}} \ & \frac{1}{2}\mathbf{u}^\top H \mathbf{u} + f^\top \mathbf{u} && \text{(17a)} \\
\text{s.t.} \ & \Psi \mathbf{M}\mathbf{u} \leq \Theta - \Psi \mathbf{n} && \text{(17b)} \\
& \underline{\mathbf{u}} \leq \mathbf{u} \leq \overline{\mathbf{u}}, && \text{(17c)}
\end{aligned}
$$

where only the first component of the minimizer sequence $\mathbf{u}^*$ is applied following a receding horizon scheme as usual in any predictive controller.

REFERENCES

[1] L. Ljung, *System Identification (2nd Ed.): Theory for the User*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1999.

[2] B. Lim and S. Zohren, "Time-series forecasting with deep learning: a survey," *Philosophical Transactions of the Royal Society A*, vol. 379, no. 2194, p. 20200209, 2021.

[3] S. H. Rudy, J. N. Kutz, and S. L. Brunton, "Deep learning of dynamics and signal-noise decomposition with time-stepping constraints," *Journal of Computational Physics*, vol. 396, pp. 483–506, 2019.

[4] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. MIT Press, 2006.

[5] D. F. Gomez, F. D. Lagor, P. B. Kirk, A. H. Lind, A. R. Jones, and D. A. Paley, "Data-driven estimation of the unsteady flowfield near an actuated airfoil," *Journal of Guidance, Control, and Dynamics*, vol. 42, no. 10, pp. 2279–2287, 2019.

[6] J. R. Salvador, D. R. Ramirez, T. Alamo, and D. Muñoz de la Peña, "Offset free data driven control: application to a process control trainer," *IET Control Theory & Applications*, vol. 13, no. 18, pp. 3096–3106, 2019.

[7] J. Roll, A. Nazin, and L. Ljung, "Nonlinear system identification via direct weight optimization," *Automatica*, vol. 41, no. 3, pp. 475–490, 2005.

[8] J. M. Manzano, D. Muñoz de la Peña, J. P. Calliess, and D. Limon, "Componentwise hölder inference for robust learning-based MPC," *IEEE Transactions on Automatic Control*, vol. 66, no. 11, pp. 5577–5583, 2021.

[9] A. D. Carnerero, D. R. Ramirez, and T. Alamo, "State-Space Kriging: A data-driven method to forecast nonlinear dynamical systems," *IEEE Control Systems Letters*, vol. 6, pp. 2258–2263, 2022.

[10] N. Cressie, "Kriging nonstationary data," *Journal of the American Statistical Association*, vol. 81, no. 395, pp. 625–634, 1986.

[11] J. Marzat and H. Piet-Lahanier, "Design of nonlinear MPC by Kriging-based optimization," *IFAC Proceedings Volumes*, vol. 45, no. 16, pp. 1490–1495, 2012.

[12] E. F. Camacho and C. Bordons, *Model predictive control*. Springer Science & Business Media, 2013.

[13] J. Salvador, T. Alamo, D. Ramirez, and D. Muñoz de la Peña, "Model predictive control of partially fading memory systems with binary inputs," *Journal of Process Control*, vol. 64, pp. 141–151, 2018.

[14] H. Wei and Y. Shi, "Mpc-based motion planning and control enables smarter and safer autonomous marine vehicles: Perspectives and a tutorial survey," *IEEE/CAA Journal of Automatica Sinica*, 2022.

[15] J. Berberich, J. Köhler, M. A. Müller, and F. Allgöwer, "Data-driven model predictive control with stability and robustness guarantees," *IEEE Transactions on Automatic Control*, vol. 66, no. 4, pp. 1702–1717, 2020.

[16] X. Wang, J. Sun, G. Wang, F. Allgöwer, and J. Chen, "Data-driven control of distributed event-triggered network systems," *IEEE/CAA Journal of Automatica Sinica*, vol. 10, no. 2, pp. 351–364, 2023.

[17] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017.

[18] D. Limon, I. Alvarado, T. Alamo, and E. F. Camacho, "MPC for tracking piecewise constant references for constrained linear systems," *Automatica*, vol. 44, no. 9, pp. 2382–2387, 2008.

[19] D. Limon, A. Ferramosca, I. Alvarado, and T. Alamo, "Nonlinear MPC for tracking piece-wise constant reference signals," *IEEE Transactions on Automatic Control*, vol. 63, no. 11, pp. 3735–3750, 2018.

[20] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge University Press, 2000.

[21] J. Shawe-Taylor and N. Cristianini, *Kernel methods for pattern analysis*. Cambridge University Press, 2004.

[22] J. P. Kleijnen, "Kriging metamodeling in simulation: A review," *European journal of operational research*, vol. 192, no. 3, pp. 707–716, 2009.

[23] G. Alfonso, A. D. Carnerero, D. R. Ramirez, and T. Alamo, "Receding horizon optimization of large trade orders," *IEEE Access*, vol. 9, pp. 63 865–63 875, 2021.

[24] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[25] G. Grimm, M. J. Messina, S. E. Tuna, and A. R. Teel, "Examples when nonlinear model predictive control is nonrobust," *Automatica*, vol. 40, no. 10, pp. 1729–1738, 2004.

[26] H. K. Khalil, *Nonlinear control*. Pearson New York, 2015, vol. 406.

[27] D. E. Seborg, T. F. Edgar, D. A. Mellichamp, and F. J. Doyle III, *Process dynamics and control*. John Wiley & Sons, 2016.

[28] G. Pannocchia and J. B. Rawlings, "Disturbance models for offset-free model-predictive control," *AIChE journal*, vol. 49, no. 2, pp. 426–437, 2003.

[29] U. Maeder, F. Borrelli, and M. Morari, "Linear offset-free Model Predictive Control," *Automatica*, vol. 45, no. 10, pp. 2214–2222, 2009.

[30] G. Pannocchia, "Offset-free tracking MPC: A tutorial review and comparison of different formulations," in *2015 European Control Conference (ECC)*, 2015, pp. 527–532.

[31] P. M. Oliveira and J. D. Hedengren, "An APMonitor temperature lab PID control experiment for undergraduate students," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2019, pp. 790–797.

A. Daniel Carnerero received his M. Eng. Degree in Industrial Engineering from the University of Seville in 2019 and the Ph.D. in Automation and Control from the University of Seville in 2022. His current research interests include GPU computing, metaheuristic optimization, model predictive control, randomized algorithms and data-driven methods.

Daniel R. Ramirez received his M.Eng. and PhD degrees in Computer Engineering from the University of Seville in 1996 and 2002, respectively. Since 2009 he has been Associate Professor of the Department of System Engineering and Automation of the University of Seville. He has authored and co-authored more than 70 technical papers in international journals and conference proceedings. His current research interests include randomized algorithms, model predictive control, data-based forecasting and soft computing techniques.

Teodoro Alamo received the M.Eng. degree in telecommunications engineering from the Polytechnic University of Madrid in 1993, and the PhD degree in telecommunications engineering from the University of Seville in 1998. He has been a Full Professor of the Department of System Engineering and Automation in the University of Seville since 2010. He was with the Ecole Nationale Superieure des Télécommunications (Télécom Paris), Paris, France, from September 1991 to May 1993. Part of his PhD work was done at RWTH Aachen, Aachen, Germany, from June to September 1995. He has co-founded the spin-off company Optimal Performance (University of Seville). He is the author or co-author of more than 200 publications, including books, book chapters, journal articles and conference proceedings. His current research interests include decision-making, model predictive control, data-driven methods, randomized algorithms, and optimization strategies.

Daniel Limon received the M.Eng. and Ph.D. degrees in electrical engineering from the University of Seville, Seville, Spain, in 1996 and 2002, respectively. From 2017 he is Full Professor of the Department of System Engineering and Automation in the University of Seville. He has been visiting researcher at the University of Cambridge and the Mitsubishi Electric Research Labs in 2016 and 2018 respectively. Dr. Limon has been a Keynote Speaker at the International Workshop on Assessment and Future Directions of Nonlinear Model Predictive Control in 2008 and Semiplenary Lecturer at the IFAC Conference on Nonlinear Model Predictive Control in 2012. He has been the Chair of the fifth IFAC Conference on Nonlinear Model Predictive Control (2015). His current research interests include model predictive control, stability and robustness analysis, tracking control and data-based control.