Sufficient Conditions for Convergent Recursive Extrapolation of qLPV Scheduling Parameters along a Prediction Horizon

Marcelo Menezes Morato, Julio E Normey-Rico, Olivier Sename

Sufficient Conditions for Convergent Recursive Extrapolation of qLPV Scheduling Parameters along a Prediction Horizon

Marcelo M Morato[1,2], Julio E. Normey-Rico[1] and Olivier Sename [2]
[1]Renewable Energy Research Group (GPER), Departamento de Automação e Sistemas (DAS),
Universidade Federal de Santa Catarina, Florianópolis, Brazil.
[2]Univ. Grenoble-Alpes, CNRS, Grenoble INP$^{\top}$, GIPSA-Lab, 38000 Grenoble, France.
$^{\top}$ Institute of Engineering, Univ. Grenoble-Alpes.

Model Predictive Control (MPC) algorithms have long been applied for nonlinear processes. In a quasi-Linear Parameter Varying (qLPV) setting, nonlinearities are included into bounded scheduling parameters, which are given as a function of endogenous variables; these scheduling parameters are *a-priori* unknown along a future prediction horizon, which complicates MPC design. To address this problem, literature points out two options: robust MPC approaches, considering the scheduling to be uncertain; or sub-optimal ones, that set values for these parameters along the horizon. With respect to the latter group, this paper proposes an extrapolation algorithm that estimates the future values of the qLPV scheduling parameters for a fixed prediction horizon of N steps; the method is based on a recursive procedure using simple Taylor expansions. Sufficient conditions for convergent extrapolation are presented with regard to the form and class of the scheduling function and the robustness of the MPC. Different benchmark examples from the literature are presented to illustrate the algorithm, which is also compared to state-of-the-art techniques.

***Index Terms*—Linear parameter-varying systems; NL predictive control; Estimation; Optimisation algorithms.**

I. Introduction

MODEL Predictive Control (MPC) is a well-established technique with many practical applications, for both linear and nonlinear systems (NMPC) [1]. The conceptual idea is to generate an optimal action at each sampling instant by solving a constrained optimisation program. This optimisation embeds performance goals, such as reference tracking or disturbance rejection, and is written in terms of a process prediction model. The MPC framework allows the designer to explicitly include the effects of input and state constraints, which is rather convenient.

For systems with Linear Time-Invariant (LTI) models, the MPC optimisation is a Quadratic Program (QP), which can be solved very fast by most modern solvers. When the model is nonlinear, the procedure becomes a Nonlinear Program (NP), being hard to solve in real-time. Nowadays, there exist some efficient NMPC algorithms [2], based on real-time iterations and gradient mappings (such as ACADO [3] and GRAMPC [4], respectively). There are also sub-optimal NMPC solutions with practical relevance [5], [6]. Despite these advances, most NMPC schemes rely on solving NPs, which is usually not possible for strict sampling periods. Thereby, the real-time embedded NMPC applications have practical impediments, as pointed out by all major systematic reviews on the topic [7].

During the last couple of years, an interesting alternative to reduce the computational toughness of NMPC algorithms has been developed, exploiting quasi-Linear Parameter Varying (qLPV) embeddings to obtained linear predictions. The LPV framework has become a very popular tool to model complex processes [8], [9]. LPV systems are linear in the state space, but nonlinear in the parameter space. Differently from LTI models, the state transition map depends on scheduling parameters (denoted ρ). These variables are bounded and known: current values $\rho(k)$ can be measured/estimated, whereas future behaviours $\rho(k+j)$ are generally not known.

As long as the Linear Differential Inclusion (LDI) property is verified, nonlinear systems can be exactly described by qLPV embeddings [10], [11]. Since many nonlinear processes satisfy LDI, the study of NMPC procedures formally conducted for qLPV models has been investigated since the late 00's [12]. A through survey of MPC methods applied to nonlinear systems represented through qLPV/LPV[1] models has been recently presented by the Authors [13].

The core concept of NMPC algorithms through qLPV embeddings is very pragmatic. Since qLPV models retain the linearity property from inputs to outputs, it becomes possible to formulate computationally-efficient design procedures. This means that the drawbacks of "full-blown" NMPC algorithms can be avoided, enabling possible real-time applications. But then, in order to solve the resulting optimisation procedure, the availability of the scheduling law becomes a crucial issue, since its evolution along the prediction horizon is unknown. In order to address this issue, literature points out to two possible routes:

(i) Robustifying the MPC, assuming that ρ is an uncertain variable along the prediction horizon, and thus solving the optimisation with respect to the worst-case performances implied by the bounds on ρ. This approach is generally referred to as worst-case, "min-max" algorithms. Yet

Corresponding author: M. M. Morato. Tel. $+55\ 48\ 3721-7596$.

[1]There is a slight difference between proper LPV models and quasi-LPV (qLPV) ones. For the first group, the scheduling parameters are exogenous, as in the case of external activation signals. In the qLPV case, the scheduling parameters are available online as a static and possibly nonlinear map of the system variables. As a results, considering MPC design, stability is typically dealt with a robust worst-case problem for the LPV case, while rendered as a NP for qLPV conditions. In this work, we consider $\rho(k) = \mathcal{F}(x(k))$.

robust, this procedure somehow looses the "essence" of LPV design, since the scheduling parameter is simply treated as an unmeasured variable. Min-Max LPV MPC has both "online-only" implementations [14], as well as implementations with offline preparations [15], [16], which solve the scheduling uncertainty problem via Linear Matrix Inequalities (LMIs).

(ii) Then, there are methods which neglect the actual trajectory for ρ along the horizon and replace it by a trajectory guess, converting the min-max QPs or the NLs into a single QP, see [5]. The scheduling trajectory guess can be considered as if ρ was kept constant along the horizon [17], [18] or found by iterative guessing mechanisms. With respect to these, we stress that the current state-of-the-art consists in Least Squares (LS) identification tools [19] and in sequential operations of the optimisation problem, using the future state predictions to compute the scheduling guess [20], [21]. The main advantage of these sub-optimal techniques is that the resulting optimisation is a QP or an SQP, which are simple enough to run online for many fast applications.

The main advantage of the robust solutions is that they are able to provide "performance certificates" for all possible values of the scheduling parameters. Nonetheless, even the "online-only" formulations often demand high computational power, thus complicating real-time applications. While possibly providing degraded performances with respect to the robust MPCs, the sub-optimal algorithms have been brought to focus since the simplicity of the resulting program enables their application for a wider variety of systems, such as embedded vehicle suspensions that operate under 5 ms samples [18].

The motivation of this paper lies within the range respectful to these sub-optimal MPC techniques. Specifically, we investigate how can the scheduling trajectory guesses be efficiently formulated and what are the conditions for their convergence? Regarding this thread, our contributions are:

1) A novel method for extrapolating qLPV scheduling parameters trajectories along a prediction horizon is proposed.
2) Sufficient conditions for the convergence of this recursive method are derived.
3) Illustrative examples, considering qLPV benchmark models from the literature, are presented to demonstrate the effectiveness and simplicity of the method. Comparisons to state-of-the-art methods are also included.

Paper organisation: The problem of recursively extrapolating the trajectory of qLPV scheduling parameters along a fixed prediction horizon is analytically formulated in Sec. II. The proposed algorithm is formulated in Sec. III. Sufficient conditions for convergence are demonstrated in Sec. IV. Several benchmark examples are provided in Sec. V. A final discussion and concluding remarks are put forth in Sec. VI.

Notation: In this work, the set of nonnegative real number is denoted by $\mathbb{R}_+$, whist the set of nonnegative integers including zero is denoted by $\mathbb{N}$. The index set $\mathbb{N}_{[a,b]}$ represents $\{i \in \mathbb{N} \,|\, a \leq i \leq b\}$, with $0 \leq a \leq b$. The identity matrix of size j is denoted as $\mathbb{I}_j$; its i-th row is denoted $I_{j,\{i\}}$. The convolution product $c(z)$ between two z-domain transfers $b(z)$ and $a(z)$ is denoted as follows: $c(z) := b(z) \circledast a(z)$. Considering a vector $v \in \mathbb{R}^{n_v}$, v_j denotes its j-th entry, with $j \in \mathbb{N}_{[1,n_v]}$. For a given sequence of scheduling parameters $P_k := \text{col}\{\rho(k+j)\}, \forall j \in \mathbb{N}_{[0,N-1]}$, $P_k\{j\}$ denotes the $(j-1)$-th entry of this column vector, i.e. $\rho(k+j)$.

Definition 1: Little-o notation
A given function $f(k)$ can be expressed as $f(k) = o(g(k))$ as $k \to \infty$ if, for every positive constant ϵ, there exists another constant β such that: $|f(k)| \leq \epsilon g(k)$ for all $k \geq \beta$.

II. Problem Statement

We consider the class of discrete-time qLPV systems:

$$x(k+1) = A(\rho(k))x(k) + B(\rho(k))u(k)\,, \quad (1)$$
$$\rho(k) = \mathcal{F}\left(x(k)\right)\,, \quad (2)$$

where $x \in \mathbb{R}^{n_x}$, $u \in \mathbb{R}^{n_u}$, $\rho \in \mathbb{R}^{n_\rho}$ and $A : \mathbb{R}^{n_\rho} \to \mathbb{R}^{n_x} \times \mathbb{R}^{n_x}$, $B : \mathbb{R}^{n_\rho} \to \mathbb{R}^{n_x} \times \mathbb{R}^{n_u}$ are continuous maps. It is implied that $\mathcal{F} : \mathbb{R}^{n_x} \to \mathbb{R}^{n_\rho}$ is a continuous map and that $\rho(k) \in \mathcal{P}\, \forall k \geq 0$, being $\mathcal{P} \subseteq \mathbb{R}^{n_\rho}$ a known set. The states are measurable for all sampling instants.

Set of Assumptions 1:

- The model matrices are affine on ρ, which means that: $A(\rho) := A_0 + \sum_{j=1}^{n_\rho} A_j \rho_j$ and $B(\rho) := B_0 + \sum_{j=1}^{n_\rho} B_j \rho_j$.
- The system is stable in closed-loop;
- The contraints on states, inputs and scheduling parameters are given by known compact sets:

$$\mathcal{X} := \left\{x \in \mathbb{R}^{n_x} \,|\, \underline{x}_j \leq x_j \leq \overline{x}_j\,, \forall j \in \mathbb{N}_{[1,n_x]}\right\},$$
$$\mathcal{U} := \left\{u \in \mathbb{R}^{n_u} \,|\, \underline{u}_j \leq u_j \leq \overline{u}_j\,, \forall j \in \mathbb{N}_{[1,n_u]}\right\}.$$
$$\mathcal{P} := \left\{\rho \in \mathbb{R}^{n_\rho} \,|\, \underline{\rho}_j \leq \rho_j \leq \overline{\rho}_j\,, \forall j \in \mathbb{N}_{[1,n_\rho]}\right\}.$$

- The qLPV scheduling map holds $\mathcal{F}(\mathcal{X}) \subseteq \mathcal{P}$.
- The variation rate of the scheduling parameters $(\rho(k+1) - \rho(k))$ is bounded.

Remark 1: The necessity of closed-loop stability is standard. For such, there must exist a feedback $u := \kappa(\cdot)x \in \mathbb{R}^{n_x}$ such that $x^+ = (A(\rho) + B(\rho)\kappa(\cdot))\,x$ is exponentially stable for all $\rho \in \mathcal{P}$. The feedback gain $\kappa(\cdot)$ could be either parameter-dependent (i.e. $\kappa(\rho)$) or constant (i.e. κ), depending on the control synthesis. For generality, we henceforth consider the MPC state-feedback as a parameter-dependent gain $\kappa(\rho)$. We also stress that, in some cases, one can only ensure that the MPC provides a closed-loop asymptotically stable system if the model is open-loop controllable. These issues should be taken into account in the control design step, which are not the focus of this paper. Refer to [13], [22] for further details on the synthesis of stable closed-loops for qLPV systems under MPC algorithms.

Remark 2: Vectors x, u and ρ are component-wise energy-bounded, in the sense of the induced $\mathcal{L}_2$ norm. It follows that $||x||_2 \leq x_{\max}$, $||u||_2 \leq u_{\max}$ and $||\rho||_2 \leq \rho_{\max}$.

Lemma 1: Assume the constraints on states, inputs and scheduling parameters ($x(k) \in \mathcal{X}, u(k) \in \mathcal{U}$, and $\rho(k) \in \mathcal{P}$, respectively) are known and valid for all sampling instants. Thus, the state deviation variable $\Delta x(k) := (x(k+1) - x(k))$ is also bounded to a compact set $\Delta\mathcal{X}$.

Proof 1: Take $\Delta x(k) := (x(k+1) - x(k)) = (A(\rho(k)) - \mathbb{I}_{n_x})\, x(k) + B(\rho(k))u(k)$. Since $(x, u, \rho) \in (\mathcal{X} \times \mathcal{U} \times \mathcal{P})$ for all instants k, it follows that $\Delta\mathcal{X} := \left\{\Delta x \in \mathbb{R}^{n_x} \,|\, \underline{\Delta x}_j \leq \Delta x_j \leq \overline{\Delta x}_j \,,\, \forall j \in \mathbb{N}_{[1,n_x]}\right\}$. The component-wise bounds $(\underline{\Delta x}_j \,,\, \overline{\Delta x}_j)$ can be determined either by interval arithmetics or optimisation. □

Regarding MPC, we consider a finite-horizon cost function:

$$J_k = \left(\sum_{j=1}^{N} \left(\ell(x(k+j), u(k+j-1))\right)\right) + V(x(k+N))$$

where $\ell(\cdot,\cdot)$ is the main cost (quadratic) and $V(\cdot)$ is a terminal offset cost. This function is formalised according to traditional tuning mechanisms, as in [21], [19]. We proceed with:

$$\ell(x, u) = x^T Q x + u^T R u \,, \tag{3}$$
$$V(x) = x^T T x \,, \tag{4}$$

being T, Q and R positive-definite weighting matrices. In some cases, the terminal weight is parameter-dependent, i.e. $T = T(\rho)$, as in [21]. For the application of the MPC scheme, the following optimisation problem is solved online, at each sampling instant k:

$$\begin{aligned} \min_{U_k} \quad & J_k \\ \text{subject to} \quad & \text{Process Model, Eq. (1)}\,, \\ & x(k+j) \in \mathcal{X}\, \forall\, j \in \mathbb{N}_{[1,N]}\,, \\ & u(k+j-1) \in \mathcal{U}\, \forall\, j \in \mathbb{N}_{[1,N]}\,, \\ & \rho(k+j) = P_k\{j\}\, \forall\, j \in \mathbb{N}_{[0,N-1]}\,, \end{aligned} \tag{5}$$

where U_k stands for the vector of control efforts inside the prediction horizon and P_k for the trajectory of the scheduling parameters along the horizon:

$$U_k = \left[\; u(k) \quad u(k+1) \quad \ldots \quad u(k+N-1) \;\right]^T , \tag{6}$$
$$P_k = \left[\; \rho(k) \quad \rho(k+1) \quad \ldots \quad \rho(k+N-1) \;\right]^T . \tag{7}$$

Due to the qLPV model in Eq. (1), the values of the scheduling parameters along the horizon, $\rho(k+j)$ for $j = 0, \ldots, N-1$, are needed to describe the future state behaviour. Note that if Eq. (2) is plugged into the optimisation, it is rendered as an NP. Nevertheless, if a guess for the values of $\rho(k+j)$ is used to replace these unknown values, the optimisation is rendered as a QP, which can be solved rapidly by standard solvers.

In previous works [13], [21], it is shown how a regular quadratically weighted cost function J_k can be rewritten in terms of P_k, and so can the predicted state trajectory X_k, as follows:

$$X_k = H(P_k)x(k) + S(P_k)U_k \,, \tag{8}$$

where $H(P_k) \in \mathbb{R}^{(n_x \times N) \times n_x}$ and $S(P_k) \in \mathbb{R}^{(n_x \times N) \times (n_u \times N)}$ are analytically presented in [21]. This means that the state predictions computed through Eq. (1) are dependent on P_k, which is the vector of the concatenated known scheduling parameters and future parameters along the horizon.

We note that the control policy found through the solution of the constrained program in Eq. (5) is conceived under a paradigm of a moving-window horizon, which slides along k as time evolves. The size of this horizon is fixed as of N samples ahead of k. This means that, at instant k, the control sequence U_k is computed considering the performances of the next N steps, and $u(k)$ is applied; at instant $k+1$, the problem $\min J_{k+1}$ is solved considering the performances from N samples ahead of $k+1$, computing U_{k+1} and the new control input $u(k+1)$ is applied. Moreover, the policy at each instant is formulated as a regular state-feedback: $u(k) = \kappa(\rho)x(k)$.

Problem Statement: Based on the previous discussion and assumptions, the problem investigated here becomes clear: to find a recursive law of the following fashion:

$$P_k = \Phi\left(P_{k-1}, \rho(k), x(k)\right) \,, \tag{9}$$

and to demonstrate that it converges to the correct scheduling trajectory values in a finite amount of samples.

The procedure in Eq. (9) generates a new extrapolation for the scheduling trajectories P_k, at instant k, based on the prior extrapolation and the new data set available ($\rho(k)$ and $x(k)$). For simplicity, we henceforth name P_k as the "scheduling sequence".

Regarding the Problem Statement, we present the novel extrapolation algorithm (in the form of Eq. (9)) in Sec. III and provide the theoretical proofs of convergence in Sec. IV.

Remark 3: The interaction between the recursive parameter trajectory estimation process (Eq. (9)) and the MPC algorithm (determining U_k by solving the optimisation program from Eq. (5) and applying $u(k)$ to the process) is as follows: (i) firstly, the current state and scheduling variables $x(k)$ and $\rho(k)$ are measured; (ii) then, the parameter trajectory is estimated through Eq. (9); (iii) the process model is used to compute the state predictions on the basis of P_k (using Eq. (8)); (iv) the optimisation procedure is solved and from its solution $U_k^\star$, the first entry $u^\star(k)$ is applied to the plant. We discuss the initialisation of the algorithm in the following Section.

III. Recursive Extrapolation Algorithm

Assumption 1: The static map $\mathcal{F}(x)$ can be approximated by the following first order Taylor expansion around $\overline{x}$:

$$\mathcal{F}(x) \approx \mathcal{F}(x)|_{\overline{x}} + \left.\frac{\partial \mathcal{F}}{\partial x}\right|_{\overline{x}} (x - \overline{x}) \,, \tag{10}$$

being $\overline{x}$ an arbitrary linearisation point. The actual function can be analytically expressed by the sum of this approximation to a residual signal w_f, which inherits the discrepancy between the real static map and its Taylor approximate:

$$\mathcal{F}(x) = \mathcal{F}(x)|_{\overline{x}} + \left.\frac{\partial \mathcal{F}}{\partial x}\right|_{\overline{x}} (x - \overline{x}) + w_f \,, \tag{11}$$

Consider Assumption 1 holds. Then, the following expression is valid, considering the linearisation at a given instant

$k+j-1$ and the increment along x to the following instant $k+j$, namely $\Delta x(k+j-1)$:

$$\begin{aligned}\mathcal{F}(x(k+j)) &= \mathcal{F}(x(k+j-1)) + w_f(k+j-1) \\ &+ \left.\frac{\partial \mathcal{F}}{\partial x}\right|_{x(k+j-1)} \Delta x(k+j-1)\,.\end{aligned} \tag{12}$$

We henceforth denote $\sigma_{k+j-1} = \left.\frac{\partial \mathcal{F}}{\partial x}\right|_{x(k+j-1)}$. Expanding the expression in Eq. (12) along the fixed prediction horizon of N steps and embedding it to Eq. (2) yields:

$$\begin{aligned}\rho(k+1) &= \rho(k) + \sigma_k \Delta x(k) + w_f(k)\,, \\ &\vdots \\ \rho(k+N-1) &= \rho(k+N-2) \\ &+ \sigma_{k+N-2}\Delta x(k+N-2) \\ &+ w_f(k+N-2)\,.\end{aligned}$$

As of Eqs. (1)-(2), $\rho(k)$ and $\Delta x(k)$ are known, whereas σ_k can be numerically evaluated on the basis of the current state measurement $x(k)$. Nevertheless, in practice, σ_{k+j} for $j \in \mathbb{N}_{[1,N-2]}$ is unknown, which requires a second assumption:

Assumption 2: For simplicity, at each sampling instant k, it is assumed that the partial derivative σ_k stays constant along the prediction horizon, i.e. $\sigma_{k+j} = \sigma_k\,, \forall\, j \in \mathbb{N}_{[1,N-2]}$.

The partial derivatives terms σ_{k+j} could be computed on the basis of the state trajectory prediction X_k (generated by the MPC algorithm). Nevertheless, this is numerically costly. Thus, we exploit Assumption 2 in order make our extrapolation procedure fast and numerically cheap, thus taking $\sigma_{k+j} = \sigma_k$. In the sequel, we show that even by using such approximation, convergence is still ensured.

Note that the expansions along the prediction horizon can be given in terms of the previous scheduling parameter value and a correction term, as follows:

$$\rho(k+j) = \rho(k+j-1) + \sigma_k \Delta x(k+j-1) + w_f(k+j-1)\,,$$

where $\rho(k+j-1)$ can be retrieved from the previous estimation. Therefore, it holds that:

$$P_k = P^\star_{k-1} + \sigma_k \Delta X^\star_k + W_k\,, \tag{13}$$

which is a recursive estimation law of the fashion in Eq. (9), as aimed. Notice that $P^\star_{k-1}$ stands for the previous scheduling trajectory estimation with the first term corrected with the known value $\rho(k)$ (known data), while $\Delta X^\star_k$ represents the state deviations along the horizon (also corrected with the known value $\Delta x(k)$). Since $\Delta X^\star_k$ represents the difference of the states over time k, this vector is computed by adapting Eq. (8). For such, we shift the control sequence U_{k-1} as a basis for U_k, with the last entry kept constant. This is, we take $\breve{U}_k = [u(k|k-1)\ \dots\ u(k+N-2|k-1)\, u(k+N-2|k-1)]^T_{n_u \times N}$. Accordingly, we obtain $\Delta X_k = H(P^\star_{k-1})\Delta x(k) + S(P^\star_{k-1})\breve{U}_k$. Lastly, we stress that W_k is a bias residual vector, which "corrupts" the extrapolation. Since this vector is unknown, we disregard it in the recursive estimation procedure, which means that Eq. (13) becomes an approximation. Nevertheless, we stress that this equation remains exact for non-null (or vanishing) residuals W_k.

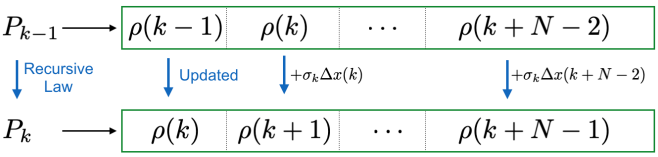

Fig. 1. Vector Shifting

Figure 1 illustrates the concept behing Eq. (13) and how the scheduling trajectory estimate from the last sample P_{k-1} can be updated in order to generate the current estimate P_k. The corrections on P_{k-1} and ΔX_k are given by:

$$P^\star_{k-1} = \lambda P_{k-1} + \nu \rho(k)\,, \tag{14}$$

$$\Delta X^\star_k = \lambda \Delta X_k + \nu \Delta x(k)\,, \tag{15}$$

with $\lambda = \begin{bmatrix} 0 & \mathbb{I} & \dots & \mathbb{I} \end{bmatrix}$ and $\nu = \begin{bmatrix} \mathbb{I} & 0 & \dots & 0 \end{bmatrix}$.

Remark 4: The dimensions of λ and ν in Eqs. (14)-(15) should be in accordance with n_ρ and n_x. We note that recursive extrapolation mechanism from Eq. (13) is not able to ensure that each entry of the scheduling trajectory estimate vector (i.e. $\rho(k+j)\,, \forall j \in \mathbb{N}_{[1,n_\rho]}$) abides to the admissibility constraints (each $\rho(k+j) \in \mathcal{P}$). Therefore, in order to provide "coherent" extrapolated parameters $\rho(k+j)$, given within the scheduling set $\mathcal{P}$, the extrapolation vector P_k is "clipped".

Remark 5: An additional forgetting factor can be included to Eqs. (14)-(15), replacing the identity matrices in λ by exponentially decaying terms, such as $\mathbb{I}e^{-k/k_{max}}$. This term would attenuate the amount of mistaken information passed from one estimate P_k to the following P_{k+1}.

Note that Assumption 2 is an approximation while the convergence property of the extrapolation mechanism has not yet been established (further details in Lemma 3), since $\sigma_{k+j} \neq \sigma_k$. Nevertheless, as the system stabilizes and the extrapolation converges, it follows that $\sigma_{k+j} \approx \sigma_k$ becomes a very reasonable approximation. This approximation may be violated more easily with larger horizons N, since the discrepancies between $\rho(k+j)$ and $\hat{\rho}(k+j)$ grow along $j \in \mathbb{N}_{[1,N-2]}$. Yet, the sliding-horizon mechanism of the MPC ensures that this imprecision has minor effects on the closed-loop performances, as illustrated by the simulation results presented in Section V.

The complete recursive estimation algorithm for qLPV scheduling trajectories is synthesised as follows:

Remark 6: The initialisation of the scheduling trajectory guess P_0 is of particular interest. At the initial point, only $x(0)$ and $\rho(0)$ are known; assumably, no control input has yet been generated, since $u(0)$ is computed on the basis of the optimisation, which requires P_0. Therefore, since the MPC is ensured to stabilize the system for all $\rho \in \mathcal{P}$ (Lemma 2), we simply use $P_0 = \text{col}\{\rho(k)\}$ (N repeated entries of $\rho(k)$). This is "the best possible candidate" for the scheduling trajectory at the initial sampling instant, and it is refined progressively as the recursive extrapolation convergences.

Recursive Extrapolation Algorithm
For every sampling instant k, loop:
$\rightarrow$ Measure the system states $x(k)$;
$\rightarrow$ Get $\rho(k)$ with Eq. (2);
$\rightarrow$ Get $x(k)$ with Eq. (1) and compute $\Delta x(k)$;
$\rightarrow$ Correct $P_{k-1} \rightarrow P^{\star}_{k-1}$ with Eq. (14);
$\rightarrow$ Compute the state deviations ΔX_k based on $P^{\star}_{k-1}$ and $\breve{U}_k$;
$\rightarrow$ Correct these deviations $\Delta X_k \rightarrow \Delta X^{\star}_k$ with Eq. (15);
$\rightarrow$ Compute the derivative σ_k;
$\rightarrow$ Compute P_k using Eq. (13) with $W_k = 0$;
$\rightarrow$ Clip each entry of the extrapolation P_k, i.e. each $\rho(k+j)$ is replaced by $\max(\underline{\rho}\,, \min(\rho(k+j)\,, \overline{\rho}))$.

IV. Sufficient Conditions for Convergence

With respect to the extrapolation algorithm, we now present sufficient conditions for its convergence. For such, the following rationale is used: if the MPC controller, based on an approximated scheduling sequence estimation, still ensures closed-loop stability, then the extrapolation P_k will converge. In the sequel, we provide a Lemma regarding the MPC state-feedback gain, and another Lemma which gives five sufficient conditions for convergence.

Lemma 2: The state-feedback MPC policy can be scheduled with respect to $\rho(k)$ in the form $u(k) = \kappa(\rho(k))x(k)$. Furthermore, there exists an upper gain κ_∞ such that $\|\kappa(\rho(k))\|_\infty \leq \kappa_\infty, \forall \rho(k) \in \mathcal{P}$.

Proof 2: Consider an MPC application for a process with constant time-invariant parameters (LTI), i.e. $x(k+1) = Ax(k) + Bu(k)$. In this case, it is trivial to show that the resulting predictive control law can be given as $u(k) = \kappa x(k)$, where the state-feedback gain depends on the system model parameters (A, B), tuning weights (T, Q, R), and prediction horizon size N. The closed-loop stability can be shown through classical Lyapunov arguments, which generate a nominal explicit feedback gain κ. Since in our study we consider a time-varying system expressed through the qLPV model in Eq. (1), the MPC can be scheduled with respect to the known scheduling parameters at each instant, i.e. $\rho(k)$, since the model matrices are dependent on this variable. Therefore, the resulting state-feedback predictive control law has also the form of $u(k) = \kappa(\rho(k))x(k)$, where the feedback gain is parameter dependent, as shown in [21]. Since the parameters are expressed within a bounded set $\mathcal{P}$, we can benefit from the polytopic representation of Eq. (1) to determine a state-feedback gain $\kappa_\infty \geq \|\kappa(\rho)\|_\infty, \forall \rho \in \mathcal{P}$ by evaluating $\kappa(\rho)$ at the vertices of the embedding polytope.

We note that the nominal state-feedback gain $\kappa(\rho(k))$ can be explicitly computed. Assume there exist a stage cost $\ell(x, u)$ and a terminal $V(x)$, as in Eqs. (3)-(4), with positive-definite weights $T(\rho)$, Q, and R. Let $Y(\rho) = (T(\rho))^{-1}$ and take $\kappa(\rho) = W(\rho)Y(\rho)$. Assume the closed-loop is stable and the MPC is recursively feasible. Then, it follows that a Lyapunov argument holds for the system, meaning that the finite-horizon MPC cost is decreasing, i.e. $J_{k+1} - J_k \leq 0, \forall k$. As detailed in [15], [16], and [21], this inequality implies in:

$$V\left((A(\rho) + B(\rho)\kappa(\rho))x\right) - V(x) \leq \ell(x, \kappa(\rho)x)\,, \forall x \in \mathcal{X}\,.$$

The above inequality can be re-stated in a parameter-dependent LMI form (see [21, Theorem 2] and Appendix A of this paper), which is solvable in a grid of points over $\rho \in \mathcal{P}$. This LMI solution provides the parameter-dependent matrices $T(\rho)$ and $W(\rho)$, which are used to generate the feedback gain $\kappa(\rho)$. This ends the proof. □

Remark 7: In order for the MPC to be robustly stable, despite the model-process uncertainties derived using all possible "wrong" (non-ideal) scheduling sequences $\breve{P}_k$, constraint tightening and robust invariant terminal set tools can be used. As in any control method, robustness comes at the expense of performance deterioration. Note, anyhow, that the proposed method provides scheduling estimates much more accurate than $\breve{P}_k$, which means less conservative performances are enable, with comparable computational load. Note that the proposed method consists basically of linear vector-wise operations, which is computationally simple. The issue of closed-loop performances of "frozen" qLPV MPC algorithms are not the focus of this paper, whereas discussed in [23].

Lemma 3: If the following Sufficient Conditions are satisfied

- (C1) The static map $\mathcal{F}(\cdot)$ is, at least, class $\mathcal{C}^1$, i.e. first-order differentiable with respect to x, for all $x \in \mathcal{X}$;
- (C2) The differentiation function σ_k is energy-bounded (induced $\mathcal{L}_2$ sense) for all k;
- (C3) The state deviation term $\Delta x(k+j)$ is energy-bounded (induced $\mathcal{L}_2$ sense) for all k;
- (C4) The qLPV system is stable in closed-loop;
- (C5) The nominal closed-loop system tolerates a prediction uncertainty $w_f(k+j-1)$ with an $\mathcal{L}_2$ norm bound $w_{\max} \leq \Delta x_{\max}$;

then, the recursively proposed extrapolation tool in Eq. (13) with $W_k = 0$ is convergent.

Proof 3: We proceed by detailing each of these five Sufficient Conditions individually: (C1): $\mathcal{F}(\cdot)$ must be at least class $\mathcal{C}^1$, so that its derivative σ_k exists for all $x(k+j) \in \mathcal{X}$. The derivative term is necessary in order for the Taylor approximation of Eq. (11) to be valid.

(C2) and (C3): Since, for simplification purposes, the recursive extrapolation is computed as if σ_{k+j} remained constant as σ_k through the prediction horizon, from the viewpoint of each sampling instant k, it must hold that $||\sigma_k||_2 \leq \sigma_{\max}$ for P_k in Eq. (13) to exist. Moreover, to construct P_k, $\Delta X^{\star}_k$ must be energy-bounded, which conversely means that each $||\Delta x(k+j)||_2 \leq \Delta x_{\max}$. This condition is also necessary for Lemma 2 to hold.

(C4): For the MPC to stabilize the system, there must exist a nominal feedback $u = \kappa(\rho)x$ such that the closed-loop dynamics are exponentially stable for all $\rho \in \mathcal{P}$, as argued in Remark 1. This is ensured if Lyapunov conditions are satisfied through adequate terminal ingredients of the MPC optimisation, computed offline, as those presented in [21]. We stress that robustness arguments against prediction uncertainties can be included through constraint tightening.

(C5): From the definition of a Taylor expansion, it follows that $w_f(k+j-1) = h_w(x(k+j-1))\Delta x(k+j-1)$. This function can be expressed through the little-o notation, using $k+j-1 = \breve{k}$: $w_f(\breve{k}) = o\left(||\Delta x(\breve{k})||_2\right)$, which translates to $||w_f(k+j-1)||_2 \leq \epsilon||\Delta x(k+j-1)||_2$, holding after a natural number of discrete-time steps $\breve{k} \geq \beta$ and a positive real constant ϵ. Since the derivative term σ_k is energy-bounded by $\sigma_{\max}$, it always holds that for any arbitrary interval on the x-plane given by $\left(x(\breve{k}) - \Delta x_{\max}, x(\breve{k}) + \Delta x_{\max}\right)$, $||w_f(\breve{k})||_2 \leq \Delta x_{\max}$, with $\beta = 1$ (one-sample deviation). Accordingly, we have $w_{\max} \leq \Delta x_{\max}$.

Next, we verify that the qLPV system when controlled through an LTI-based prediction model MPC maintains stability despite the variations caused by the scheduling parameter and the Taylor expansion error w_f. We must ensure, thus, that an $M-\Delta$ robust stability condition is verified with respect to the bound on the uncertainties implied by w_f.

For such, we consider that there exists a specific form on how to describe the LPV matrices: the system is assumed affine (refer to Set of Assumptions 1). At any given instant, we can freeze qLPV system as a perturbed LTI model, computed upon the instantaneous $\rho(k)$ and uncertainty upon it, denoted $w_f(k)$. This is, using the uncertainty inputs:

$$\begin{aligned} u_{\Delta_x}(z) &= \overbrace{(A_1 w_f(z)) \circledast x(z)}^{\Delta_x(z)x(z)}, \\ u_{\Delta_u}(z) &= \underbrace{(B_1 w_f(z)) \circledast u(z)}_{\Delta_u(z)u(z)}, \end{aligned}$$

we are able to obtain the following state-space description:

$$\begin{aligned} x(k+1) &= A_n x(k) + B_n u(k) + \overbrace{(u_{\Delta_x}(k) + u_{\Delta_u}(k))}^{u_\Delta(k)}, \\ A_n &= A_0 + A_1\rho(k), \quad B_n = B_0 + B_1\rho(k). \end{aligned}$$

Consider two "uncertainty outputs": $y_{\Delta_x}(k) = x(k)$ and $y_{\Delta_u}(k) = u(k)$, being $y_\Delta(k) = \text{diag}\{y_{\Delta_x}(k), y_{\Delta_u}(k)\}$. The corresponding "frozen" LTI model is expressed in the z-domain as follows:

$$\begin{aligned} x(z) &= G_n(z)u(z) + G_\Delta(z)\left(u_{\Delta_x}(z) + u_{\Delta_u}(z)\right), \\ G_n(z) &= \left(z\mathbb{I}_{n_x} - A_n\right)^{-1} B_n, \; G_\Delta(z) = \left(z\mathbb{I}_{n_x} - A_n\right)^{-1}. \end{aligned}$$

Then, we use Lemma 2 to state the control policy and add a fictive input $r(z)$ to the control input so that input-output stability can be verified:

$$\begin{aligned} \|x(z)\|_\infty \leq & \overbrace{\| \left(\mathbb{I}_{n_x} - G_n(z)\kappa_\infty\right)^{-1} G_n(z)\kappa_\infty}^{T_n(z)} r(z) \\ & + \left(\mathbb{I}_{n_x} - G_n(z)\kappa_\infty\right)^{-1} G_\Delta(z) u_{\Delta_x}(z) \\ & + \underbrace{\left(\mathbb{I}_{n_x} - G_n(z)\kappa_\infty\right)^{-1} G_\Delta(z)}_{T_\Delta(z)} u_{\Delta_u}(z)\|_\infty, \end{aligned}$$

The Linear Fractional Transformation (LFT) of this frozen system is, for $N := F_\ell(P, \kappa_\infty)$:

$$\left\| \begin{bmatrix} y_\Delta(z) \\ x(z) \end{bmatrix} \right\|_\infty \leq \|N(z)\|_\infty \left\| \begin{bmatrix} u_\Delta(z) \\ r(z) \end{bmatrix} \right\|_\infty.$$

$$N(z) = \left[\begin{array}{c|c} \overbrace{\begin{pmatrix} T_\Delta(z) & T_\Delta(z) \\ T_\Delta(z)\kappa_\infty & T_\Delta(z)\kappa_\infty \end{pmatrix}}^{M(z)} & \begin{pmatrix} T_n(z) \\ \kappa_\infty + T_n(z)\kappa_\infty \end{pmatrix} \\ \hline \begin{pmatrix} T_\Delta(z) & T_\Delta(z) \end{pmatrix} & T_n(z) \end{array}\right].$$

Finally, we can write $u_\Delta(k) = \Delta(\cdot)y_\Delta(k)$, with:

$$\begin{aligned} ||\Delta_x(\cdot)||_\infty &= \frac{||A_1 w_f \circledast x||_\infty}{x_{\max}}, \\ ||\Delta_u(\cdot)||_\infty &= \frac{||B_1 w_f \circledast u||_\infty}{u_{\max}}. \end{aligned}$$

Due to the definition of the Taylor expansion and, as constructed, due to the fact that $||w_f(k+j-1)||_2 \leq \Delta x_{\max} \leq \epsilon||\Delta x(k+j-1)||_2$, holding for all $k+j-1 \geq 1$ and every positive real constant ϵ, the convolution products are upper bounded as follows:

$$||w_f \circledast x|| \leq \Delta x_{\max} x_{\max}, \; ||w_f \circledast u|| \leq \Delta x_{\max} u_{\max}.$$

Due to this fact, it follows that:

$$\begin{aligned} ||\Delta(\cdot)||_\infty &= \left\| \begin{matrix} \Delta_x(\cdot) & 0 \\ 0 & \Delta_u(\cdot) \end{matrix} \right\|_\infty \\ &\leq \left\| \begin{matrix} A_1 & 0 \\ 0 & B_1 \end{matrix} \right\|_\infty \begin{bmatrix} \frac{||w_f \circledast x||_2}{x_{\max}} & 0 \\ 0 & \frac{||w_f \circledast u||_2}{u_{\max}} \end{bmatrix} \\ &\leq \left\| \begin{matrix} A_1 & 0 \\ 0 & B_1 \end{matrix} \right\|_\infty \Delta x_{\max}. \end{aligned}$$

Note that, $||M(z)||_\infty \leq ||\kappa_\infty T_\Delta(z)||_\infty$, which means that $||M(z)||_\infty \leq ||\kappa_\infty(\mathbb{I}_{n_x} - G_n(z)\kappa_\infty)^{-1}G_\Delta(z)||_\infty$. Thence, the stability condition is very direct: simply checking the following inequality, for all $\rho \in \mathcal{P}$ and $x \in \mathcal{X}$ (where κ_∞ is determined according to Lemma 2):

$$||\kappa_\infty T_\Delta(z)||_\infty \leq \frac{1}{||\Delta(\cdot)||_\infty}. \tag{16}$$

This concludes the proof. □

Remark 8: Condition (C5) in Lemma 3 is essential. As long as the closed-loop system is robustly stable despite the residual term w_f, we can demonstrate the convergence property by showing that $\lim_{k\to+\infty} w_f(k+j) \to 0$ and that $\sigma_{k+j} = \sigma_k$, $\forall j \in \mathbb{N}_{[1,N-2]}$. Assume the process is stable in closed-loop. Thus, it holds that $\lim_{k\to\infty} x(k+j) = x(k+j-1)$. Then, take $w_f(k+j-1) = \mathcal{F}(x(k+j)) - \mathcal{F}(x(k+j-1)) - \sigma_{k+j-1}\Delta x(k+j-1)$. It directly follows from (C5) that $\sigma_{k+j-1} = \sigma_k$, $\forall j \in \mathbb{N}_{[1,N-2]}$. Thus, $\lim_{k\to\infty} \mathcal{F}(x(k+1)) = \lim_{k\to\infty} \mathcal{F}(x(k))$ and $\lim_{k\to\infty} \Delta x(k) = 0$. Finally, $\lim_{k\to\infty} w_f(k) = -\lim_{k\to\infty} \sigma_k \Delta x(k) \to 0$, which conversely ensures that the scheduling sequence extrapolation P_k indeed converges to the real scheduling sequence.

Remark 9: In order to verify that the proposed method ensures a convergent extrapolation, (C5) can be checked through the following Algorithm, used to verify inequality (16). We note that the computation of the compact set $\Delta\mathcal{X}$ from Lemma 1 requires optimisation or interval arithmetics, while the feedback gain κ_∞ is derived using LMIs (as provided in Lemma 2 and also expressed in [24].

Checking Sufficient Condition (C5)
For every $\rho \in \mathcal{P}$:
$\rightarrow$ Compute the feedback gain κ_∞ via Lemma 2, solving the optimisation for each $x(k) \in \mathcal{X}$ and each $\rho(k) \in \mathcal{P}$;
$\rightarrow$ Compute the nominal matrices A_n and B_n;
$\rightarrow$ Compute the closed-loop nominal model $T_\Delta(z)$;
$\rightarrow$ Compute the uncertainty bound $||\Delta(\cdot)||_\infty$;
$\rightarrow$ Compute $||\kappa_\infty T_\Delta(z)||_\infty$ and verify inequality (16).

V. Illustrative Simulations

In this Section, we provide application results of the proposed recursive extrapolation tool. Three different case studies are considered: a numeric example, a semi-active suspension system and a pendubot system. All results were obtained with the aid of *Matlab*, *Yalmip* and *Gurobi* solver, performed on a 2.4 GHz, 8 GB RAM Macintosh computer. Through the first two examples in the sequel, the state-feedback MPCs in the form of $u = \kappa(\rho)x$ are synthesised with identity Q and R weights and a parameter-dependent $T(\rho)$ derived from the solution of the LMIs in [21, Theorem 2].

The first case study is chosen because it exhibits a time-varying derivative σ_{k+j-1}. Therefore, it serves to demonstrate that even if Assumption 2 is violated, if the five conditions from Lemma 3 are satisfied, the algorithm still ensures convergence.

The second and third case studies are chosen from the literature because LPV MPC techniques have considered these models (see [19] and [20], respectively). The second case presents some load disturbances which meddle with the stabilisation of the process, while the third has a larger number of states (six). Moreover, both these cases have constant or null derivative terms σ_{k+j}, differing from the first.

In order to illustrate the advantages of the proposed mechanism, we also include comparisons of our method in Examples A and B, considering the state-of-the-art technique of iterative SQPs from [21] and the simplified gain-scheduled/ "frozen-guess" MPC approach from [23].

Note that all the following curves are of discrete-time systems. For simplicity, nevertheless, we opt to exhibit the discrete time sample as line curves generated by the linear interpolation between these samples.

A. Numeric Example

Consider a qLPV system operating at a sampling rate of 1000 Hz. It is controlled by an MPC loop which operates within 1 ms. The system model is $x(k+1) = A(\rho(k))x(k) + B(\rho(k))u(k)$, with:

$$
\begin{aligned}
A(\rho(k)) &= \begin{bmatrix} -0.5(1+\rho(k)) & 0 \\ 0 & -0.3(1+\rho(k)) \end{bmatrix} \\
B(\rho(k)) &= \begin{bmatrix} (1+\rho(k)) \\ 2(1+\rho(k)) \end{bmatrix} \\
\rho(k) &= \mathcal{F}(x(k)) = \sin\left(\begin{bmatrix} 1 & 1 \end{bmatrix} x(k)\right), \\
\sigma_{k+j-1} &= \begin{bmatrix} 1 & 1 \end{bmatrix} \cos\left(\begin{bmatrix} 1 & 1 \end{bmatrix} x(k+j-1)\right).
\end{aligned}
$$

This system has box-type constraints as are those in the Set of Assumptions 1, with: $\underline{x} = -[0.5\,0.3]^T$, $\overline{x} = [0.5\,0.4]^T$, $\underline{u} = -0.025$, $\overline{u} = 0.025$, $\underline{\rho} = -1$, $\overline{\rho} = 1$. Both state deviation and scheduling parameter deviation variables are energy-bounded.

With respect to the five sufficient conditions given in Section IV, they are all verified:

1) A sine function is class $\mathcal{C}^\infty$ w.r.t. its domain;
2) A cosine is always energy-bounded;
3) Due to the box-type constraints on x, $||\Delta x(\cdot)|| \leq 0.8$;
4) Indeed, the closed-loop qLPV system is stable under the corresponding MPC algorithm;
5) Algorithm IV verifies the inequality from (C5) for all $\rho \in \mathcal{P}$: it holds that $\sup_{\rho(k)\in\mathcal{P}} ||\kappa_\infty T_\Delta(z)||_\infty = 0.1651$, while $(||\Delta(\cdot)||_\infty)^{-1} = 0.6667$.

The simulation results are presented in the sequel, which show that the recursive extrapolation converges within roughly 5 samples, i.e. 5 ms. The control horizon is taken as $N = 30$ samples. The total simulation run comprises 0.2 s; the system is perturbed by a disturbance signal at $t = 0.1$ s. Fig. 2 shows the stabilisation of the states and the predictive control policy. Most importantly, Fig. 3 shows the extrapolation of P_k at different instants and the actual qLPV scheduling trajectories, demonstrating that convergence of P_k is indeed verified.

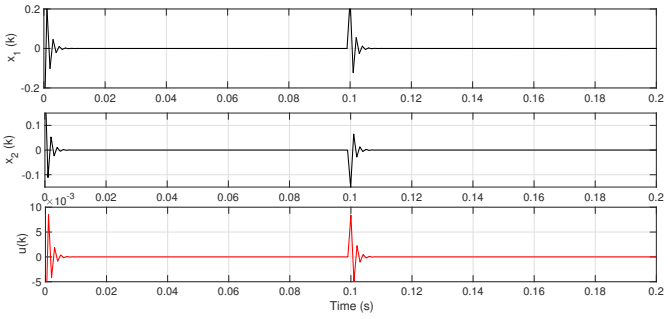

Fig. 2. Numerical Example: State Stabilisation (black line) and Control Signal (red line).

For illustration purposes, the proposed method is compared to the two other qLPV MPC approaches widely used in the literature. The first is the simplistic "frozen-guess" method [23], for which $\breve{P}_k = \rho(k)\mathbf{1}_{1\times N}$. The second is the iterative QP from [21], for which the MPC QP is solved multiple times, with P_k is iteratively taken as $f_\rho(X_k)$. The latter also has convergence guarantees, while requiring the solution of several QPs per sampling period, as well as the application of a vector-wise nonlinear proxy $f_\rho(\cdot)$.

These MPC methods are synthesised with the same tuning weights. We compare the obtained performances in terms of normalised RMS (NRMS) indexes for each state trajectory $x_j(k)$, which are presented in Table I. As it can be seen, the resulting closed-loop performances are very similar with the proposed method and the one by [21], both slightly superior than the frozen guess method (baseline indexes). Nevertheless, we must stress that the MPC with the proposed extrapolation mechanism can operate four times faster than the state-of-the-art SQP scheme, since it does not require to evaluate any nonlinear vector-wise operation online, neither multiple QPs.

In order to further illustrate the convergence of the scheduling sequence estimates, Fig. 4 compares the extrapolations

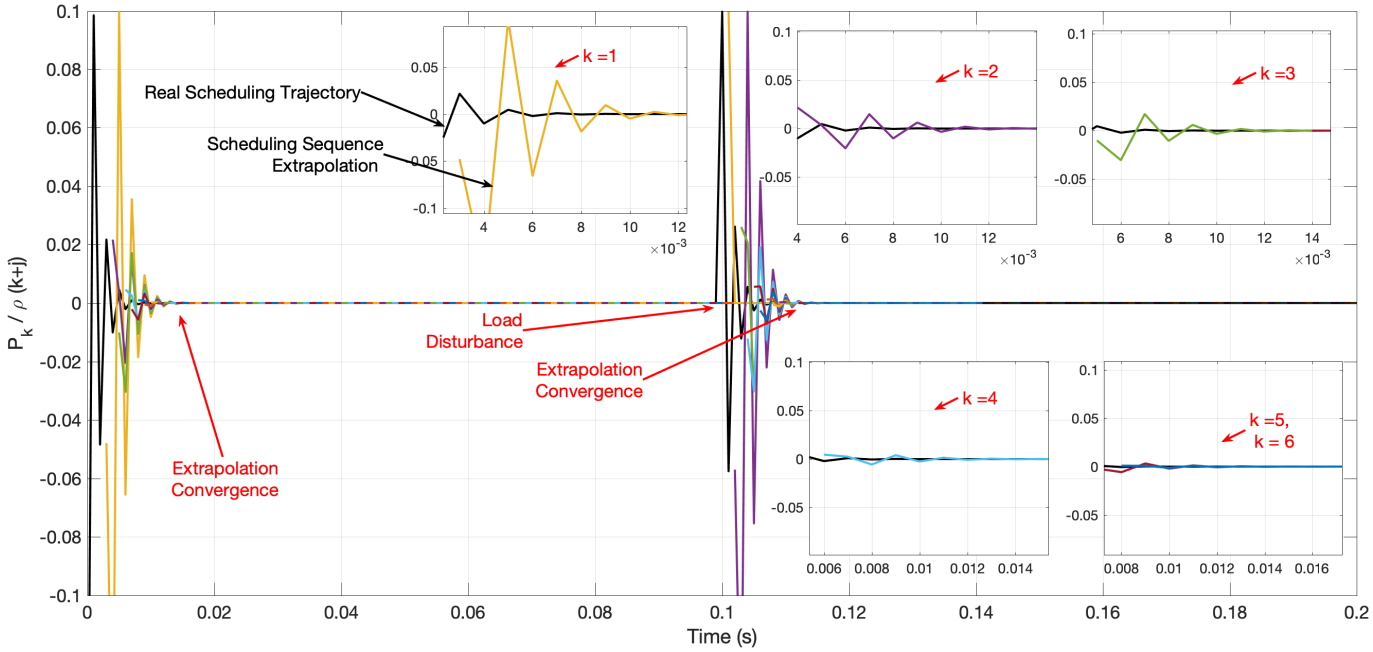

Fig. 3. Numerical Example: Scheduling Trajectory Extrapolation and Convergence. The real scheduling parameter trajectory is presented in bold black line. The coloured lines represent the extrapolated scheduling sequence at different time instants.

derived with the proposed method against the estimated with the iterative mechanism of [21]. In this Fig., P_k (real trajectory) and $\hat{P}_k$ (estimate) are given for the first six simulation samples. Note that $\hat{P}_0$ is an initial guess. Clearly, both methods converge in roughly the same number of samples. The discrepancy between $\hat{P}_5$ and P_5 are slightly smaller with the approach from [21], but this advantage comes at the expense of more computational cost. Nevertheless, we note that these discrepancies can be reduced with adequate forgetting factors (see Remark 5), such that the prior estimate data P_{k-1} have less effects on the following estimate P_k.

Table I evidences an important feature of the proposed method, which we highlight. The additional computational time t_c required to solve the recursive extrapolation mechanism is of $0.01\,\text{ms}$ (on average), with respect to the "frozen guess" MPC. Even with such minor additional computational load (of roughly 8%, in this case), the closed-loop performances are much enhanced. We stress that the total computational load of an MPC scheme operating together with the proposed extrapolation algorithm is very close to that of a QP. This is due to the fact that the operation of Eq. (13) consists only of linear vector-wise operations, whose numerical toughness depends linearly on the size of the prediction horizon N and on the number of number of scheduling parameters n_ρ.

In the case of systems with a higher number of states, the numerical load required by proposed estimation law will represent an even smaller ration w.r.t. the load required by the MPC QP, which grows exponentially with n_x. This is a very relevant advantage of the proposed method, since the state-of-the-art iterative SQP mechanism from [21] grows exponentially with n_x and with N, while also being proportional to n_{iter}, which is the average number of iterations of the QPs, per sampling period. Thereby, for systems with an elevated number of states, the iterative SQP framework from [21] may easily violate the sampling period threshold of real-time applications ($t_c < T_s$), while the proposed scheme may not, since it will require, basically, the computational time needed for a single QP.

TABLE I
NUMERICAL EXAMPLE: PERFORMANCE EVALUATION.

Method	NRMS$\{x_1\}$	NRMS$\{x_2\}$	t_c
Frozen Guess [23]	$100\,\%$	$100,\%$	0.124 ms
Iterative SQPs [21]	$84.62\,\%$	$89.86\,\%$	0.632 ms
Proposed	$83.84\,\%$	$89.86\,\%$	0.134 ms

B. Semi-Active Suspension System

As a second study case, we consider a semi-active suspension system, represented by a quarter-car qLPV benchmark model from the literature [23]. The dynamics comprise the displacements and velocities of a car's chassis and wheel (four states). The scheduling parameter is the suspension deflection velocity. This system operates with a $5\,\text{ms}$ sampling period, and is disrupted by road bumping (load disturbances). The MPC horizon is of $N = 10$; the system has box-type constraints on states and inputs.

The model is $x(k+1) = A(\rho(k))x(k) + B_1(\rho(k))u(k) + B_2 w(k)$, where $w(k)$ are the road disturbances and $\rho(k) = (x_2(k) - x_4(k))$. Further details are given in references [23], [25].

All sufficient conditions from Lemma 3 are satisfied:

1) The difference $(x_2(k) - x_4(k))$ is a linear operator and thus class $\mathcal{C}^\infty$;
2) $\sigma_k = [0\,1\,0\,-1]$ is energy-bounded;

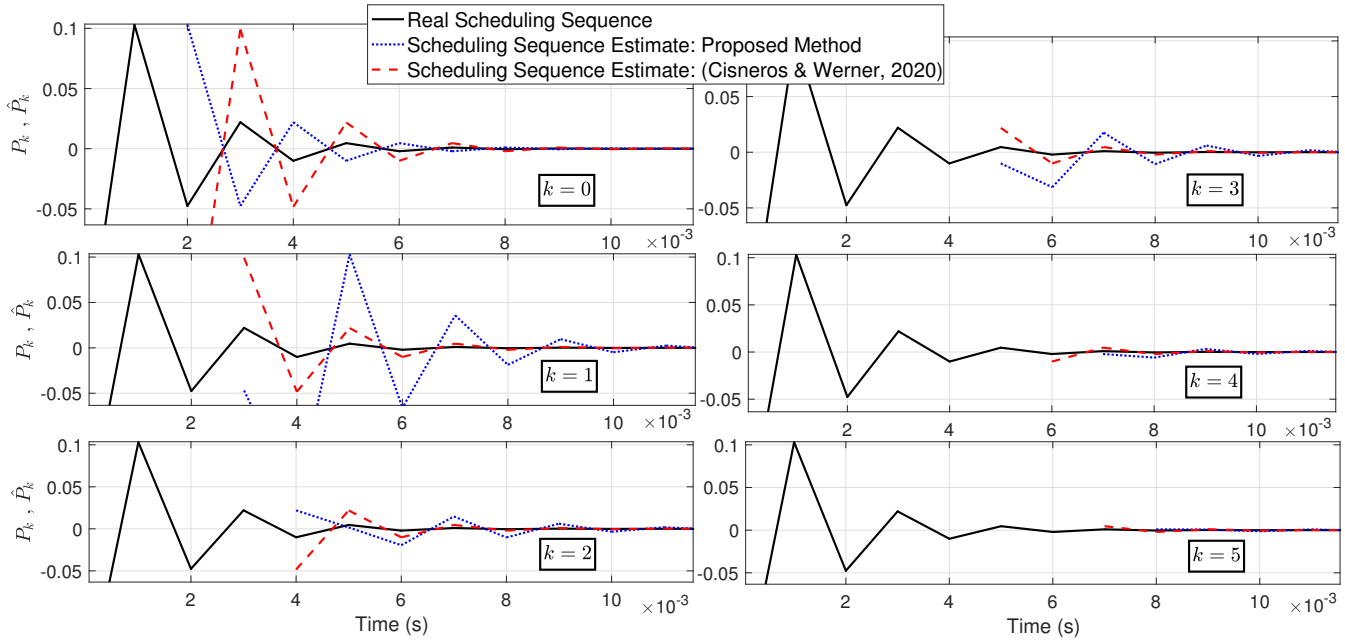

Fig. 4. Numerical Example: Scheduling Trajectory Estimation Convergence - Proposed Method and [21]. The real scheduling parameter trajectory is presented in bold black lines, the estimates with the proposed method in dotted blue lines, and the estimates with the method from [21] in dashed blue lines.

3) The box-type constraints ensure energy bounds on Δx;
4) The qLPV model is stable in closed-loop;
5) Algorithm IV verifies the inequality from (C5) for all $\rho \in \mathcal{P}$: it holds that $\sup_{\rho(k)\in\mathcal{P}} ||\kappa_\infty T_\Delta(z)||_\infty = 15.54$, while $(||\Delta(\cdot)||_\infty)^{-1} = 66.73$.

We consider a simulation scenario of 10 s with ± 5 mm bump-like road disturbances at three different instants. The extrapolation algorithm converges within 25 samples, i.e. 0.125 ms. The average computational stress is of 0.048 ms, on average. Fig. 5 shows the stabilisation of the states to the origin, the predictive control policy and the road profile disturbances, while Fig. 6 shows the real scheduling trajectories P_k and the corresponding recursive extrapolation estimates $\hat{P}_k$ at different sampling instantes over the simulation run.

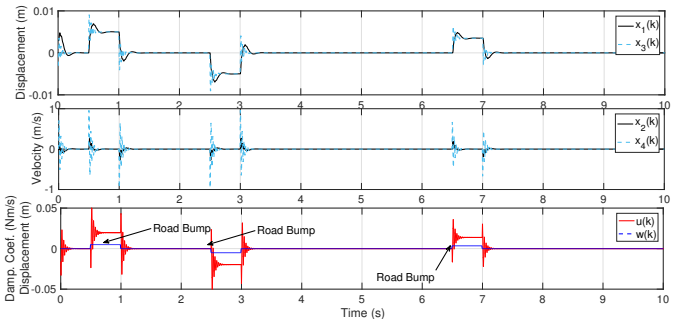

Fig. 5. Vehicle Suspension Example: State Stabilisation (bold, black and dashed, light blue lines), Control (bold, red line), and Road Disturbances (bold, blue line).

In order to corroborate the comparison discussions from the first example, the proposed method is once again tested against the qLPV MPC approaches from [21] (iterative SQPs) and [23] (gain-scheduled/"frozen-guess" method). The MPCs are synthesised with the same tuning weights. In this example, due to larger number of states ($n_x = 4$) than the prior, we compare the obtained performances in terms of the normalised RMS index for the stage cost trajectory $\ell(x(k), u(k))$, which are presented in Table II. Firstly, we stress that the frozen-guess mechanism already obtains good driving performances by itself, as argued in [23]. Nevertheless, as show in Table II, the resulting closed-loop performances are enhanced with the proposed method as with the method by [21], both with superior indexes than the gain-scheduled baseline result. As indicated in previous discussions, the proposed extrapolation mechanism yield almost negligible computational time with respect to the time required by the QP. Furthermore, the iterative SQPs approach almost violates the sampling time constraint of 5 ms, taking over ten times more than the MPC coupled with the proposed estimation procedure (on average). This occurs singe the QPs grow exponentially with the number of states, while the extrapolation does not.

TABLE II
SEMI-ACTIVE SUSPENSION EXAMPLE: PERFORMANCE EVALUATION.

Method	NRMS$\{\ell(x,u)\}$	t_c
Frozen Guess [23]	100 %	0.370 ms
Iterative SQPs [21]	56.85 %	3.795 ms
Proposed	68.06 %	0.377 ms

C. Pendubot

A final example is provided. Consider the six-states pendubot benchmark system from [20]. This inverted pendulum has two arms at rotating angles measured with respect to the vertical axis. A motor is connected to the first arm and acts as the actuator in this system. The control goal is to stabilize

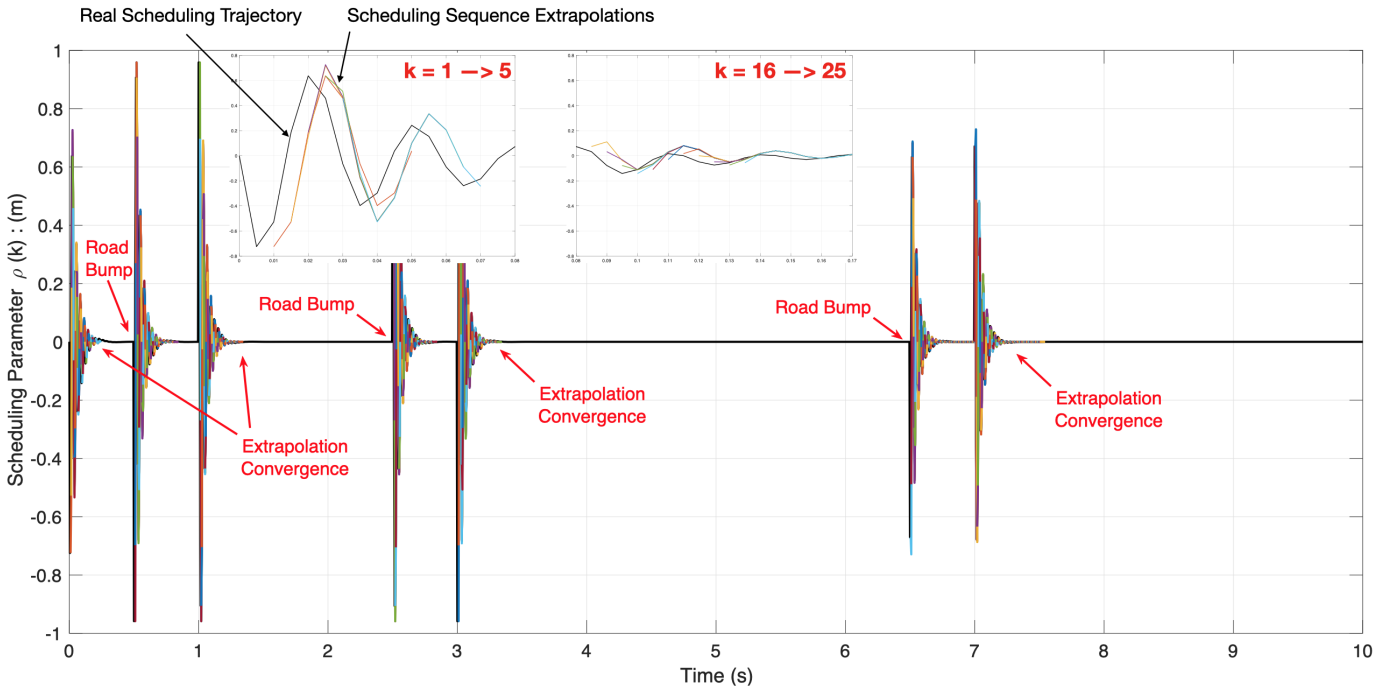

Fig. 6. Vehicle Suspension Example: Scheduling Trajectory Extrapolation. The real scheduling parameter trajectory is presented in bold black line. The coloured lines represent the extrapolated scheduling sequence at different time instants.

the system at a vertical up-up equilibrium (origin). The dynamics of this system may be disturbed by the occurrence of unexpected torques against the rotating arms.

This pendubot is represented by a discrete-time qLPV model in the form of Eq. (1), with an additional $w(k)$ term summed to the state transition map, which represents the torque disturbance. The system operates under a sampling period T_s of $10\,\text{ms}$. The model exhibits two state-related scheduling parameters: $\rho(k) = [x_1(k)\,,\, x_3(k)]^T$. This system is controlled by a sub-optimal MPC algorithm with horizon $N = 40$ steps. All matrices, parameters and MPC weights (T, Q, and R) are given in [20]. Once again, all five sufficient conditions from Lemma 3 are satisfied. It holds that $\sup_{\rho(k)\in\mathcal{P}} ||\kappa_\infty T_\Delta(z)||_\infty = 0.0845$, while $(||\Delta(\cdot)||_\infty)^{-1} = 0.2140$.

In order to illustrate this the closed-loop behaviour of this process under the action of an MPC algorithm based on the proposed recursive extrapolation procedure, we consider a simulation run of $3\,\text{s}$. In this scenario, the initial conditions are non-null and a load disturbance torque that occurs at $t = 2\,\text{s}$, which requires the controller to stabilise the pendubot at the up-up equilibrium (steering the state trajectories to the origin) twice. In Figure 7, we show the trajectories of this system, considering the stabilisation of the first four states (positions and velocities, accelerations are suppressed for simplicity), as well as the generated predictive control policy. Complementary, Figure 8 presents the extrapolation of the scheduling trajectory[2], which clearly convergences rapidly.

The average computational time required to evaluate the recursive extrapolation mechanism is of $2\,\text{ms}$. As of this, the total control law (extrapolation and QP solution) is evaluated within the $10\,\text{ms}$ sampling period threshold. We also note that the extrapolation convergence is achieved within $0.5\,\text{ms}$ (both due to initial conditions and due to the torque disturbance). The obtained performances are coherent with those presented in [20]. Evidently, the proposed solution enables a wide variety of applications.

[2]Only one of the scheduling parameters is shown, for simplicity; similar results were obtained for the other parameter.

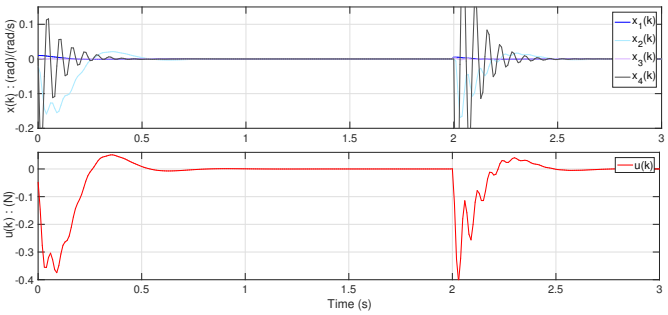

Fig. 7. Pendubot Example: State Stabilisation (black, light blue, light purple, and gray lines) and Control Signal (red line).

D. Discussion

As evidenced by these previous results, it is clear that if the five sufficient conditions from Lemma 3 are satisfied, the proposed recursive extrapolation algorithm indeed converges. Furthermore, the proposed method is able to enhanced MPC performances with respect to a gain-scheduled/"frozen guess" approach, while maintaining computational load close to that of a single QP. The recursive extrapolation mechanism resides on simple linear operators, with numerical toughness growing

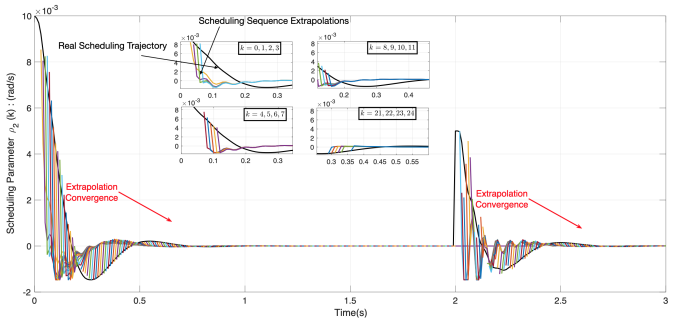

Fig. 8. Pendubot Example: Scheduling Trajectory Extrapolation. The real scheduling parameter trajectory is presented in bold black line. The coloured lines represent the extrapolated scheduling sequence at different time instants.

linearly with the prediction horizon size N and the number of scheduling variables n_ρ. The state-of-the-art SQP mechanism from [21] also provides convergence guarantees, while with a numerical toughness that grows exponentially with N and with the number of system states n_x. Therefore, for larger order real-time systems, the proposed method may be much more suitable, since it requires much less computational load.

We stress that the proposed tool was analysed for a given class of qLPV systems, with affine parameter dependency. Nonetheless, other classes could also be considered, such as polynomial forms $A(\rho(k)) = A_0 + A_1\rho(k) + A_2\rho^2(k) + \dots$ or LFT forms. The only difference would be the computation of the $u_\Delta(k)$ terms in the fifth sufficient condition (C5), which must embed the uncertainty term on the scheduling parameter caused by some bad estimation at a given instant. For the second-order polynomial case, we would have:

$$u_{\Delta_x}(k) \quad = \quad A_1 w_f(k) + A_2(w_f(k) + 2\rho(k)w_f(k))w_f(k)\,.$$

VI. Conclusions and Perspectives

This paper presented a new method on how to extrapolate the scheduling parameters of qLPV systems along a fixed prediction horizon, for MPC purposes. The method resides in a simple and fast recursive law, which only needs the evaluation of a partial derivative computation at each sampling instant. Furthermore, five simple-to-verify sufficient conditions are presented for the convergence of the proposed method. Finally, the algorithm is validated with three different qLPV benchmark tests, via simulation, demonstrating both effectiveness and convergence properties. The proposed method is compared to the state-of-the mechanism of estimating scheduling parameters through SQPs (looping the MPC multiple times), showing equivalent estimates and similar convergence rate.

The proposed tool can certainly serve for the design of fast qLPV MPC loops, which can be based on a single constrained QPs derived on the basis of the extrapolation guesses for the scheduling trajectories. By doing so, the nonlinearities from the model prediction contraints are removed. Accordingly, the generated MPC enable the application for fast systems, with strict sampling periods, which is impossible for robust min-max MPCs due to their excessive computational load demand.

We also note that the obtained closed-loop performances are enhanced with the proposed mechanism, in comparison to a gain-scheduled ("frozen guess") approach and to the state-of-the-art SQP method. For further works, the Authors plan on providing necessary conditions of the algorithm's convergence with respect to the class (affine, polytopic, LFT, etc.) of the LPV model. Furthermore, the validity of Assumption 2 will be assessed in the sense of the Lebesgue size of the domain of attraction of the MPC.

Acknowledgment

This work has been supported by CNPq project $304032/2019-0$. Marcelo M. Morato thanks Profs. Hector Bessa, Marcelo de Lelis and Daniel Coutinho (UFSC) for their helpful discussions and comments.

Appendix A
Parameter-dependent State-feedback MPC

The following Theorems are provided in order to complement Lemma 2. They are carefully demonstrated in [21].

Theorem 1 (Terminal Ingredients): Suppose the MPC law is given by $u = K(\rho)x$, considering a terminal state set given by $\mathbf{X}_f(\rho)$ and a terminal cost $V(x,\rho)$. Then, input-to-state stability is ensured if the following conditions hold $\forall \rho \in \mathcal{P}$:
(C1) The origin lies in the interior of $\mathbf{X}_f(\rho)$;
(C2) Any consecutive state to x, in closed-loop given by $(A(\rho) + B(\rho)K(\rho))\,x$ lies within $\mathbf{X}_f(\rho)$;
(C3) The discrete Lyapunov equation is verified within this invariant set, this is, $\forall\, x \in \mathbf{X}_f(\rho)$ and $\forall\, \rho \in \mathcal{P}$ and $\forall\, \delta\rho \in \delta\mathcal{P}$: $V\left((A(\rho+B(\rho)K(\rho))\,x, \rho+\delta\rho\right) - V\left(x,\rho\right) \leq -x^T Q x - x^T(K(\rho)^T R K(\rho)x$.
(C4) The image of the nominal feedback lies within the admissible control domain: $K(\rho)x \in \mathcal{U}\,, \forall \rho \in \mathcal{P}$.
(C5) The terminal set $\mathbf{X}_f(\rho)$ is a subset of $\mathcal{X}$. Assuming that the initial solution of the MPC problem is feasible, then, the MPC is recursively feasible, stabilising the state origin.

Theorem 2 (Parameter-dependent Terminal Ingredients): Conditions (C1)-(C5) of Theorem 1 are satisfied if there exist a symmetric parameter-dependent positive definite matrix $T(\rho) : \mathbb{R}^{n_p} \to \mathbb{R}^{n_x \times n_x}$ and a parameter-dependent rectangular matrix $W(\rho) : \mathbb{R}^{n_p} \to \mathbb{R}^{n_u \times n_x}$, with $Y(\rho) = (T(\rho))^{-1} > 0$ and $W(\rho) = K(\rho)Y(\rho)$, such that LMIs (17)-(19) hold for all $\rho \in \mathcal{P}$ and $\delta\rho \in \delta\mathcal{P}$, with $i \in \mathbb{N}_{[1,n_u]}$ and $j \in \mathbb{N}_{[1,n_x]}$, under the minimisation of $\log\det\{Y(\rho)\}$.

$$\left[\begin{array}{cc|cc} Y(\rho) & \star & \star & \star \\ A_\pi(\rho) & Y(\rho+\delta\rho) & \star & \star \\ \hline Y(\rho) & 0 & Q^{-1} & \star \\ W(\rho) & 0 & 0 & R^{-1} \end{array}\right] \geq 0\,, \quad (17)$$

$$\left[\begin{array}{c|c} \overline{u}_i^2 & I_{n_u,\{i\}}W(\rho) \\ \hline \star & Y(\rho) \end{array}\right] \geq 0\,, \quad (18)$$

$$\left[\begin{array}{c|c} \overline{x}_j^2 & I_{n_x,\{j\}}Y(\rho) \\ \hline I_{n_x,\{j\}}^T Y^T(\rho) & Y(\rho) \end{array}\right] \geq 0\,, \quad (19)$$

$$A_\pi(\rho) \quad = \quad (A(\rho)Y(\rho) + B(\rho)W(\rho))\,. \quad (20)$$

The proof of the prior is based on Schur complements and on the positive-definiteness of T, Q and R. Matrices $Y(\rho)$ and $W(\rho)$ are generated, which are used to compute the MPC terminal ingredients $V(\cdot)$ and $\mathbf{X}_f$ such that input-to-state stability and recursively feasibility properties are guaranteed.

REFERENCES

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

[2] S. Gros, M. Zanon, R. Quirynen, A. Bemporad, and M. Diehl, "From linear to nonlinear MPC: bridging the gap via the real-time iteration," *International Journal of Control*, vol. 93, no. 1, pp. 62–80, 2020.

[3] R. Quirynen, M. Vukov, M. Zanon, and M. Diehl, "Autogenerating microsecond solvers for nonlinear MPC: a tutorial using ACADO integrators," *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 685–704, 2015.

[4] T. Englert, A. Völz, F. Mesmer, S. Rhein, and K. Graichen, "A software framework for embedded nonlinear model predictive control using a gradient-based augmented lagrangian approach (GRAMPC)," *Optimization and Engineering*, vol. 20, no. 3, pp. 769–809, 2019.

[5] Y. Zhang, S. Li, and L. Liao, "Near-optimal control of nonlinear dynamical systems: A brief survey," *Annual Reviews in Control*, vol. 47, pp. 71–80, 2019.

[6] K. M. M. Rathai, M. Alamir, O. Sename, and R. Tang, "A parameterized NMPC scheme for embedded control of semi-active suspension system," *IFAC-PapersOnLine*, vol. 51, no. 20, pp. 301–306, 2018.

[7] F. Allgöwer and A. Zheng, *Nonlinear model predictive control*. Birkhäuser, 2012, vol. 26.

[8] C. Hoffmann and H. Werner, "A survey of linear parameter-varying control applications validated by experiments or high-fidelity simulations," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 2, pp. 416–433, 2014.

[9] H. S. Abbas, R. Toth, M. Petreczky, N. Meskin, and J. Mohammadpour, "Embedding of nonlinear systems in a linear parameter-varying representation," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 6907–6913, 2014.

[10] J. S. Shamma and M. Athans, "Gain scheduling: Potential hazards and possible remedies," *IEEE Control Systems Magazine*, vol. 12, no. 3, pp. 101–107, 1992.

[11] D. J. Leith and W. E. Leithead, "Survey of gain-scheduling analysis and design," *International journal of control*, vol. 73, no. 11, pp. 1001–1025, 2000.

[12] A. Bachnas, R. Tóth, J. Ludlage, and A. Mesbah, "A review on data-driven linear parameter-varying modeling approaches: A high-purity distillation column case study," *Journal of Process Control*, vol. 24, no. 4, pp. 272–285, 2014.

[13] M. M. Morato, J. E. Normey-Rico, and O. Sename, "Model predictive control design for linear parameter varying systems: A survey," *Annual Reviews in Control*, 2020.

[14] Y. Su, K. K. Tan, and T. H. Lee, "Tube-based quasi-min-max output feedback MPC for LPV systems," in *Preprints of the 8^{th} IFAC Symposium on advanced control of chemical processes*. IFAC, 2012, pp. 10–13.

[15] M. Jungers, R. P. Caun, R. C. L. F. Oliveira, and P. L. D. Peres, "Model predictive control for linear parameter varying systems using path-dependent lyapunov functions," *IFAC Proceedings Volumes*, vol. 42, no. 2, pp. 97–102, 2009.

[16] P. Bumroongsri, "An offline formulation of MPC for LPV systems using linear matrix inequalities," *Journal of Applied Mathematics*, vol. 2014, 2014.

[17] E. Alcalá, V. Puig, and J. Quevedo, "LPV-MPC control for autonomous vehicles," in *Proceedings of the 3^{th} IFAC Workshop on Linear Parameter Varying Systems, Eindhoven, The Netherlands, Nov. 4-6, 2019*, 2019.

[18] M. M. Morato, O. Sename, and L. Dugard, "LPV-MPC fault tolerant control of automotive suspension dampers," *IFAC-PapersOnLine*, vol. 51, no. 26, pp. 31–36, 2018.

[19] M. M. Morato, J. E. Normey-Rico, and O. Sename, "Novel qLPV MPC design with least-squares scheduling prediction," in *Proceedings of the 3^{th} IFAC Workshop on Linear Parameter Varying Systems, Eindhoven, The Netherlands, Nov. 4-6, 2019*, 2019.

[20] P. S. G. Cisneros and H. Werner, "Wide range stabilization of a pendubot using quasi-LPV predictive control," in *Proceedings of the 3^{th} IFAC Workshop on Linear Parameter Varying Systems, Eindhoven, The Netherlands, Nov. 4-6, 2019*, 2019.

[21] ——, "Nonlinear model predictive control for models in quasi-linear parameter varying form," *International Journal of Robust and Nonlinear Control*, vol. 30, no. 10, pp. 3945–3959, 2020.

[22] J. Hanema, R. Tóth, and M. Lazar, "Stabilizing non-linear model predictive control using linear parameter-varying embeddings and tubes," *IET Control Theory & Applications*, vol. 15, no. 10, pp. 1404–1421, 2021.

[23] M. M. Morato, J. E. Normey-Rico, and O. Sename, "Sub-optimal recursively feasible linear parameter-varying predictive algorithm for semi-active suspension control," *IET Control Theory & Applications*, vol. 14, no. 18, pp. 2764–2775, 2020.

[24] P. S. G. Cisneros and H. Werner, "A dissipativity formulation for stability analysis of nonlinear and parameter dependent MPC," in *Annual American Control Conference*. IEEE, 2018, pp. 3894–3899.

[25] M. M. Morato, M. Q. Nguyen, O. Sename, and L. Dugard, "Design of a fast real-time LPV model predictive control system for semi-active suspension control of a full vehicle," *Journal of the Franklin Institute*, 2018.

Marcelo Menezes Morato is a PhD Candidate in Automation and Systems Engineering at UFSC/Brazil and Automation and Production and UGA/France. Currently, he holds a substitute professor position at the Automation and Systems Department at UFSC. He is ad-hoc referee to multiple journals, including International Journal of Electrical Power & Energy Systems, Control Engineering Practice, and Renewable Energy. He received the 2019 Green Talents award from the German government for young high-potential scientists due to his relevant research in sustainability.

Julio Elias Normey-Rico received his Ph.D. degree from University of Seville, Spain, in 1999. He is currently a Full Professor of the Dept. of Automation and Systems Engineering in Federal University of Santa Catarina (UFSC), in Brazil and head researcher of Renewable Energies Research Team (GPER/UFSC), lead research group in this topic in Latin America. He is the director of several research partnerships with energy industries and international cooperation agreements (Argentina, Uruguay, Spain, Chile and Italy). He is the author of over 260 conference and journal papers, and published four book chapters and a full textbook on Control of Dead-Time Processes (Springer). He has supervised over 60 PhD/MSc. candidates. He was associated Editor of Control Engineering Practice from 2007 to 2018 and Editor of the International Renewable Energy Congress since 2014. Since 2000 he integrates the NOC of several national conferences related with automatic control, and recently, he was the General Chair of the IFAC Symposium DYCOPS 2019.

Olivier Sename received a Ph.D. degree from Ecole Centrale Nantes in 1994. He is now Professor at the Institut Polytechnique de Grenoble within GIPSA-lab. His main research interests include Linear Parameter Varying systems and automotive applications. He is the (co-)author of 2 books, 60 international journal papers, and more than 200 international conference papers. He was the General Chair of the IFAC Joint Conference SSSC-TDS-FDA 2013, of the 1st IFAC Workshop on Linear Parameter Varying Systems 2015 and he was the IPC Chair of the 2nd IFAC Workshop LPVS 2018. He has led several industrial (Renault, Volvo Trucks, JTEKT, Delphi) and international (Mexico, Italy, Hungary) collaboration projects. He has supervised 32 Ph.D. students.