Morato *et al*. *Complex Eng Syst* 2021;1:5
DOI: 10.20517/ces.2021.09

Complex Engineering Systems

Research Article

Open Access

A qLPV Nonlinear Model Predictive Control with Moving Horizon Estimation

Marcelo Menezes Morato[1,2], Emanuel Bernardi[3], Vladimir Stojanovic[4]

[1]Departamento de Automação e Sistemas, Universidade Federal de Santa Catarina, Florianópolis 88040-900, Brazil.
[2]Univ. Grenoble-Alpes, CNRS, Grenoble INP (Institute of Engineering, Univ. Grenoble-Alpes), GIPSA-Lab, Grenoble 38000, France.
[3]Applied Control & Embedded Systems - Research Group *(AC&ES-RG)*, Universidad Tecnológica Nacional, San Francisco 2400, Argentina.
[4]The Faculty of Mechanical and Civil Engineering in Kraljevo, Department of Automatic Control, Robotics and Fluid Technique, University of Kragujevac, Dositejeva 19, Kraljevo 36000, Serbia.

Correspondence to: Marcelo Menezes Morato, DAS/CTC/UFSC, Trindade, Caixa Postal 476, Florianópolis-SC, 88040-900, Brazil. marcelomnzm@gmail.com

How to cite this article: Morato MM, Bernardi E, Stojanovic V. A qLPV Nonlinear Model Predictive Control with Moving Horizon Estimation. *Complex Eng Syst* 2021;1:5. https://dx.doi.org/10.20517/ces.2021.09

Received: 27 Aug 2021 **First Decision:** 15 Aug 2021 **Revised:** 22 Sep 2021 **Accepted:** 28 Sep 2021 **Published:** 15 Oct 2021

Academic Editor: Hamid Reza Karimi **Copy Editor:** Huan-Liang Wu **Production Editor:** Huan-Liang Wu

Abstract

This paper presents a Model Predictive Control (MPC) algorithm for Nonlinear systems represented through quasi-Linear Parameter Varying (qLPV) embeddings. Input-to-state stability is ensured through parameter-dependent terminal ingredients, computed offline via Linear Matrix Inequalities. The online operation comprises three consecutive Quadratic Programs (QPs) and, thus, is computationally efficient and able to run in real-time for a variety of applications. These QPs stand for the control optimization (MPC) and a Moving-Horizon Estimation (MHE) scheme that predicts the behaviour of the scheduling parameters along the future horizon. The method is practical and simple to implement. Its effectiveness is assessed through a benchmark example (a CSTR system).

Keywords: Nonlinear Model Predictive Control, Quasi-Linear Parameter Varying Systems, Moving Horizon Estimation, Linear Matrix Inequalities, CSTR

1 INTRODUCTION

Model Predictive Control (MPC) is a very powerful control method, with widespread industrial application. The core idea of MPC[1] is simple enough: a process model is used to predict the future output response of the process; then, at each instant, the control law is found through the solution of an online optimization problem, which is written in terms of the model, the process constraints and the performance goals. For the case of processes represented by Linear Time-Invariant (LTI) models, MPC is translated as a constrained Quadratic Programming Problem (QP), which can be evaluated in real-time by the majority of standard solvers.

Extra attention should be payed to the fact that the theoretical establishment MPC was especially consolidated after the proposition of "terminal ingredients", which served to demonstrate robust stability and recursive feasibility properties[2]. These properties are enabled when some conditions with respect to a terminal stage cost $V(\cdot)$ and to a terminal constraint $\mathbf{X}_f$ are verified. Essentially, the terminal set must be robust positively invariant for the controlled system, the stage cost must be $\mathcal{K}$-class lower bounded and the terminal cost $V(\cdot)$ should be $\mathcal{K}$-class upper bounded and Lyapunov-decreasing (it must decay along the horizon).

For many years, MPC was mostly seen in the process industry, regulating usually slower applications (with longer sampling periods). This was mainly due to the fact that the inherent optimization procedures were excessively costly (numerical-wise) and became impractical for real-time systems.

Nonlinear MPC (NMPC) algorithms yield complex optimization procedure, with exponential growth of the numerical burden. Nevertheless, the majority of system is indeed nonlinear and, thus, literature has devoted special attention to feasible NMPC design since the 00's[3]. Originally, NMPC algorithms were hardly able to run in real-time[4], but recent research effort has focused to a great extent on ways to simplify or approximate, usually through Gauss-Newton, Lagrangian or multiple-shooting discretization approaches[5], the online Nonlinear Programming Problem (NP) in order to make it viable for fast, time-critical processes. Some of these faster NMPC algorithms run within the range of a few milliseconds, resorting to solver-based solutions (as in ACADO[6] or GRAMPC[7] algorithms) or GPU-based schemes[8,9].

Parallel to these approximated methods, another research route is now expanding to address the complexity drawback of "full-blown" NMPC strategies: using quasi-/Linear Parameter Varying (qLPV/LPV) model structures to embed the nonlinear dynamics, as in[10], and thus facilitate the online optimization. Since LPV models retain linearity properties through the input/output channels, the optimization can be reduced to the complexity of a QP. A recent survey[11] details the vast possibilities of issuing NMPC through LPV structures. The basic requirement of these methods is that the nonlinearities must respect the Linear Differential Inclusion (LDI) property[12,13], in such a way that they can be embedded into a qLPV realisation, appropriately "hidden" in scheduling parameters ρ.

Instead of using a moving-window linearization strategy to yield fast NMPCs with time-varying models[14], or of using approximated solutions of the NP iterations[6], this paper follows the lines of the qLPV embedding framework, which allows for an exact description of the nonlinear system and, thereby, no time-consuming linearization or Jacobian computation needs to take place. As previously evidenced[11], these qLPV methods are able to use the scheduling proxy $\rho(k) = f(x(k), u(k))$ to compute the process predictions rapidly. In fact, these methods have recently been shown[15] to outrank (or perform equivalently as) fast NMPC solvers, such as ACADO. Some of these recent development are further detailed:

- Some works[16,17] opt to consider a frozen/constant guess for the scheduling parameters along the future horizon and ensure, through the use of terminal ingredients, that the trajectories are sufficiently regulated, despite the uncertainty along the horizon;
- Morato et al.[18] propose a method to determine an educated estimation for scheduling variables using a

recursive Least-Squares procedure. A similar procedure is applied in [19]. The main drawback is that the results could be sub-optimal, meaning that local minima found through their QPs/ Sequential QPs (SQPs), which may not ensure sufficient performances.

- The most prominent results are those reported in the recent works by Cisneros & Werner [20–22]. The original idea [20] is to iteratively use the prediction for the future state trajectories (output of the QP) to compute a guess of the scheduling parameters, using the nonlinear proxy $\rho(k+j) = f(x(k+j))$. The method was extended [21] to reference-tracking and shown to yield a Second-order Cone Program (2^{nd}OCP) formulation for the resulting NMPC, which is easier to solve than an NP. The formulation was further smoothed in the most novel reference [22], wherein the procedure is split into an offline preparation part, using Linear Matrix Inequalities (LMIs) to compute a robust positively invariant terminal set, and an online residing solely in re-iterating SQPs.

1.1 Contributions and rganization

oAs detailed in the prequel, the topic of NMPC through qLPV embedding has been studied by a handful of papers and deserves further attention. It seems that the development of these strategies can surely be established as a competitive category for nonlinear MPC design, regarding time-critical applications.

Pursuing this matter and motivated by the previous discussion, this paper proposes an alternative formulation to the recent algorithm by Cisneros and Werner [22]. In their work, the nonlinear proxy has to be evaluated online w.r.t. to the future state evolution prediction originated through the QP. The alternative procedure proposed herein relies on approximating the nonlinear proxy by a time-varying auto-regressive function, whose parameters are found through another QP, based on a Moving-Horizon Estimation (MHE) method. This alternative is able to slightly boost the numerical performances of the whole algorithm, which only needs to evaluate three QPs to find the control law.

Accordingly, the contributions presented are the following:

- An alternative formulation for NMPC is proposed: using qLPV embeddings, the MPC operates together with an MHE layer, which estimates the future behaviour of the scheduling parameters.
- The convergence of the algorithm is demonstrated.
- A benchmark example is used to demonstrate the effectiveness of the proposed scheme, in terms of performance and numerical burden.

Regarding organization, this paper is structured as follows. Idn the next Section, the preliminaries and formalities are presented, especially regarding how nonlinear processes can be embedded into a qLPV representation through LDI. Moreover, the problem setup regarding MPC applied to such qLPV model is presented. Furthermore, the proposed MHE-MPC formulation and the discussion about stability and an offline LMI-solvable remedy for the computation of the terminal ingredients is addressed. Lastly, Section simulation results and general conclusions are drawn.

1.2 Basic efinitions

Definition 1. *Nonlinear Programming Problem*
Consider an arbitrary real-valued nonlinear function $f_c(x_c)$. A Nonlinear Programming Problem determines the vector x_c that minimizes $f_c(x_c)$ subject to $g_i(x_c) \leq 0$, $h_j(x_c) = 0$ and $x_c \in \mathcal{X}_c$, where g_i and h_j are also nonlinear.

Definition 2. *Quadratic Programming Problem*
A Quadratic Programming Problem (or simply Quadratic Problem) is a linearly constrained mathematical optimization problem of a quadratic function. A QP is a particular type of NP. The quadratic function may be defined with respect to several variables, all of which may be subject to linear constraints. Considering a vector

$c \in \mathbb{R}^{n_c}$, a symmetric matrix $Q_c \in \mathbb{R}^{n_c \times n_c}$, a real matrix $A_{ineq} \in \mathbb{R}^{m_c \times n_c}$, a real matrix $A_{eq} \in \mathbb{R}^{m_c \times n_c}$, a vector $b_{ineq} \in \mathbb{R}^{m_c}$ and another vector $b_{eq} \in \mathbb{R}^{m_c}$, the goal of a QP *is to determine the vector $x_c \in \mathbb{R}^{n_c}$ that minimizes a regular quadratic function of form $\frac{1}{2}\left(x_c^T Q_x x_c + c^T x_x\right)$ subject to constraints $A_{ineq} x_c \leq b_{ineq}$ and $A_{eq} x_c = b_{eq}$. The solution x_c to this kind of problem is found by many solvers seen in the literature, based on Interior Point algorithms, quadratic search, etc.*

1.3 Notation

In this work, the set of non-negative real number is denoted by $\mathbb{R}_+$, whist the set of non-negative 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 $\mathbb{I}_j$; $\text{col}\{a\,,\, b\,,\, c\}$ denotes the vectorization (collection) of the entries and $\text{diag}\{v\}$ denotes the diagonal matrix generated with the line vector v.

The value of a given variable $v(k)$ at time instant $k+i$, computed based on the information available at instant k, is denoted as $v(k+i|k)$.

$\mathcal{K}$ refers to the class of positive and strictly increasing scalar functions that pass through the origin. A given function $f : \mathbb{R} \to \mathbb{R}$ is of class $\mathcal{K}$ if $f(0) = 0$ and $\lim_{\xi \to +\infty} f(\xi) \to +\infty$. A real-valued scalar function $\phi : \mathbb{R}_+ \to \mathbb{R}_+$ belongs to class $\mathcal{K}_\infty$ if it belongs to class $\mathcal{K}$ and it is radially unbounded (this is $\lim_{s \to +\infty} \phi(s) \to +\infty$. A function $\beta : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}_+$ belongs to class $\mathcal{KL}$ if, for each fixed $m \in \mathbb{R}_+$, $\beta(\cdot, m) \in \mathcal{K}$ and, for each fixed $s \in \mathbb{R}_+$, $\beta(s, \cdot)$ is non-increasing and holds for $\lim_{m \to +\infty} \beta(s, m) = 0$.

$\mathcal{C}^n$ denotes the set of all compact convex subsets of $\mathbb{R}^n$. A convex and compact set $X \in \mathcal{C}^n$ with non-empty interior, which contains the origin, is named a PC-set. A subset of $\mathbb{R}^n$ is denoted a polyhedron if it is an intersection of a finite number of half spaces. A polytope is defined as a compact polyhedron. A polytope can be analogously represented as the convex hull of a finite number of points in $\mathbb{R}^n$. A hyperbox is a convex polytope where all the ruling hyperplanes are parallel with respect to their axes.

Finally, consider two sets $A \subset \mathbb{R}^n$ and $B \subset \mathbb{R}^n$. The Minkowski set addition is defined by $A \oplus B := \{a+b \,|\, a \in A\,,\, b \in B\}$, while the Pontryagin set difference is defined by $A \ominus B := \{a \,|\, a \oplus B \subseteq A\}$. The cartesian product between two sets is defined as $\mathcal{A} \times \mathcal{B}$.

2. PRELIMINARIES

In this Section, we detail how nonlinear processes can be described under a qLPV formalism; we also present some other formalities.

2.1 The Nonlinear System and its qLPV Embedding

We consider the following generic discrete-time nonlinear system:

$$\begin{cases} x(k+1) &= f\left(x(k), u(k)\right)\,, \\ y(k) &= f_y\left(x(k), u(k)\right)\,, \end{cases} \tag{1}$$

where $k \in \mathbb{N}$ represents the sampling instant, $x : \mathbb{N} \to \mathcal{X} \subset \mathbb{R}^{n_x}$ represents the system states, $u : \mathbb{N} \to \mathcal{U} \subset \mathbb{R}^{n_u}$ is the vector of control inputs and $y : \mathbb{N} \to \mathcal{Y} \subseteq \mathbb{R}^{n_y}$ stands for the measured outputs of the process.

We begin by characterizing this process, which should satisfy the following key Assumptions:

Assumption 1. *The admissible zone for the states is given by a 2-norm upper bound on each entry x_j, this is:*

$$\mathcal{X} \;:=\; \left\{x \in \mathbb{R}^{n_x} \,|\, ||x_j||^2 \leq \bar{x}_j\,,\, \forall\, j \in \mathbb{N}_{[1,n_x]}\right\}. \tag{2}$$

Assumption 2. *The admissible region for the control inputs is given by a 2-norm upper bound on each entry u_j, this is:*

$$\mathcal{U} \;:=\; \left\{u \in \mathbb{R}^{n_u} \,|\, ||u_j||^2 \leq \overline{u}_j \,,\, \forall\, j \in \mathbb{N}_{[1,n_u]}\right\}. \tag{3}$$

Assumption 3. *The nonlinear maps $f : \mathcal{X}\times\mathcal{U} \to \mathcal{X}$ and $f_y : \mathcal{X}\times\mathcal{U} \to \mathcal{Y}$ are continuous and continuously differentiable with respect to x, i.e. class C^∞*

Assumption 4. *This nonlinear system is controllable in terms of x and y through the input trajectory u.*

To represent any nonlinear system, as the one in Eq. (1), under a qLPV formalism, this last Assumption must be verified, since it is the LDI property that furnishes the settings for such representation.

The LDI property is as follows: suppose that, for each x, u and y and for every sampling instant k, there exists a matrix $H(x,u,k) : \mathcal{X}\times\mathcal{U}\times\mathbb{N} \to \mathcal{H}$ such that

$$\begin{bmatrix} f\,(x(k),u(k)) \\ f_y\,(x(k),u(k)) \end{bmatrix} \;=\; H(x,u,k)\begin{bmatrix} x(k) \\ u(k) \end{bmatrix}, \tag{4}$$

where $\mathcal{H} \subseteq \mathbb{R}^{(n_x)\times(n_x+n_u)}$ is the set within which the LDI property holds. Then, when there exists a matrix $H(\cdot)$ that verifies Eq. (4), the nonlinear model from Eq. (1) can be equivalently expressed as:

$$\begin{cases} x(k+1) &=& A(\rho(k))x(k) &+& B(\rho(k))u(k)\,, \\ y(k) &=& C(\rho(k))x(k) &+& D(\rho(k))u(k)\,, \\ \rho(k) &=& f_\rho(x(k),u(k)) &\in& \mathcal{P}\,, \end{cases} \tag{5}$$

which is a qLPV formulation where $f_\rho : \mathcal{X}\times\mathcal{U} \to \mathcal{P} \subset \mathbb{R}^{n_p}$ represents the endogenous nonlinear function for the scheduling parameters. Note that $\rho(k)$ is bounded and known online at each instant k, but generally unknown for any future instant $k+j$, $\forall j \in \mathbb{N}_{[1,\infty]}$.

We consider that the qLPV scheduling parameters have bounded rates of variations, this is: $\rho(k+1) = \rho(k) + \partial\rho(k+1)$, being $\partial\rho \in \partial\mathcal{P}$, $\forall\, k$ their variation rates. This is very reasonable for any practical application. To assume that ρ varies arbitrary implies in quite conservative control synthesis [23].

3. PROBLEM SETUP

Regarding the qLPV embedded model in Eq. (5), we proceed by detailing how MPC can be applied to regulate and control this system. For the simplify of the reference tracking demonstrations, we drop the input-output energy transfer, i.e. $D(\rho(k)) = 0$. We note that the processes with $D(\rho(k)) \neq 0$ can still be dealt with the proposed method, we no additional drawbacks.

The essential idea behind MPC is to consider a quadratic finite-horizon functional cost, which embeds the performance objectives of the system within this given horizon. The implementation resides in minimizing this cost with respect to a control signal sequence, using a model of the system in order to make predictions for the future variable values along the horizon. The optimization also includes the operational constraints of the process variables (admissibility region). Generically, we consider the following steady-state reference tracking performance cost:

$$J(x,u,k) \;=\; V\left(x(k+N_p|k)\right) + \sum_{i=0}^{N_p-1} ||x(k+i|k) - x_r||_Q^2 + \sum_{i=1}^{N_p} ||u(k+i-1|k) - u_r||_R^2\,, \tag{6}$$

where Q and R are positive definite weighting matrices and the pair $(x_r\,,\,u_r)$ defines a known admissible steady-state reference target for the nonlinear system. The optimization cost J considers a prediction horizon of N_p steps and a positive terminal stage value $V\left(x(k+N_p|k)\right) > 0$.

The MPC framework considers a moving-window strategy. Therefore, at each sampling instant k, since $x(k)$, and $\rho(k)$ are known, the corresponding optimization problem is solved, which gives the solution $U_k \in \mathbb{R}^{n_u \times N_p}$. This solution constitutes the following sequence of control inputs

$$U_k = \begin{bmatrix} u(k|k) & \dots & u(k+N_p-1|k) \end{bmatrix}^T, \tag{7}$$

whose first input $u(k|k) = I_1 U_k$ is applied to the process. Then, the horizon slides forward and the procedure is updated. The complete optimization, at each sampling instant k, is given as follows:

$$\begin{aligned} &\min_{U_k} J(x,u,k) \\ \text{s.t.} \quad &\overbrace{x(k+i+1|k) = A(\rho(k+i))x(k+i|k) + B(\rho(k+i))u(k+i|k)}^{\text{LPV Process Model}}, \\ &\overbrace{u(k+i-1|k) \in \mathcal{U}}^{\text{Control Input Admissibility}} \forall i \in \mathbb{N}_{[1,N_p]}, \\ &\overbrace{x(k+i|k) \in \mathcal{X}}^{\text{Admissible Process Operation}} \quad \forall i \in \mathbb{N}_{[1,N_p]}, \\ &\overbrace{x(k+N_p|k) \in \mathbf{X}_f}^{\text{Terminal Set Constraint}}, \end{aligned} \tag{8}$$

where $\mathbf{X}_f$ and $V(\cdot)$ are the terminal ingredients, combined to ensure recursive feasibility of the algorithm (see Section *Stability and Offline Preparations*).

Due to Eq. (8), it follows that the future values for the qLPV scheduling variables $\rho(k+i)$ are not known for any $i \geq 1$. At each instant k, the optimization operates based on the knowledge of $x(k)$ and $u(k)$, which can be used to compute $\rho(k)$ through the nonlinear qLPV proxy $f_\rho(\cdot)$. One could easily include this proxy into the optimization, making it also subject to $\rho(k+i) = f_\rho(x(k+i), u(k+i))$ together with the process model, but this would convert Eq. (8) into a Nonlinear Programming Problem, which is associated with numerical complexity issues (as previously discussed).

The NP execution is computationally unattractive[22] because of this general nonlinear dependence of the predicted states on the control inputs and on previous states. Therefore, following the lines of previous works[19,22], this paper pursues a fast implementation of the LPV MPC optimization procedure in Eq. (8), which means that we do not seek to analytically include the nonlinear qLPV scheduling proxy $f_\rho(\cdot)$ to the optimization, but rather to provide values for the complete evolution of the scheduling parameters along the prediction horizon, as if they were known (thus detaching the nonlinear dependency). This is, we aim to solve Eq. (8) based on $x(k)$, $\rho(k)$ and on the future "scheduling sequence" vector $P_k \in \mathbb{R}^{n_p \times N_p}$, being

$$P_k = \begin{bmatrix} \rho(k) & \dots & \rho(k+N_p-1|k) \end{bmatrix}^T. \tag{9}$$

If the actual evolution of the scheduling sequence is as gives P_k, the MPC ensures perfect regulation. Furthermore, it is formulated as a Quadratic Programming Problem, which can be tackled for many time-critical applications with modern solvers. In fact, we approximate the NP solution by one which resides in a "guess" for the scheduling sequence P_k, which attractively converges to the actual value of this vector as the procedure iterates. The solution to estimate P_k is based on a Moving Horizon Estimation algorithm, which is further detailed in Section *The MHE-MPC Mechanism*.

We must proceed by providing some complementary Assumptions regarding this qLPV MPC optimization problem setup. For such, we denote $X_k \in \mathbb{R}^{n_x \times N_p}$ as the evolution of the state values along the prediction

horizon, this is:

$$X_k \quad = \quad \left[\; x(k+1|k) \quad \dots \quad x(k+N_p|k) \; \right]^T . \tag{10}$$

Assumption 5. *The qLPV scheduling proxy is set-wise and vector-wise applicable, this is, it holds as* $f_\rho(\mathcal{X}, \mathcal{U})$ *and also as* $f_\rho(X_k, U_k)$. *The first operation stand for the application of* $f_\rho(\cdot)$ *to the bounds of each entry set, while the later stands for the application of* $f_\rho(\cdot)$ *to each sample of the entry vectors.*

Assumption 6. *The application of the scheduling proxy to the admissible zone for the states and inputs is a subset of the scheduling set, this is:*

$$f_\rho \left(\mathcal{X}, \mathcal{U} \right) \quad \subset \quad \mathcal{P} . \tag{11}$$

Assumption 7. *The admissible region* $\mathcal{X} \times \mathcal{U}$ *is a subset of the image of the inverse of the scheduling proxy domain, being* $f_\rho(\cdot)$ *bijective. This means that the inverse of the scheduling proxy always maps admissibility pairs* $(x\,,\,u)$ *from admissible scheduling variables* ρ. *This is mathematically expressed as follows:*

$$\{ \mathcal{X} \times \mathcal{U} \} \quad \subset \quad Im \left\{ f_\rho^{-1} \left(\mathcal{P} \right) \right\} . \tag{12}$$

From the viewpoint of each sampling instant k, the scheduling sequence can be directly evaluated as:

$$P_k \quad = \quad f_\rho(X_k^\star, U_k) \, , \tag{13}$$

where $X_k^\star$ comprises the instantaneous states and the state evolution X_k until $x(k+N_p-1|k)$:

$$X_k^\star \quad = \quad \left[\; x(k) \quad \dots \quad x(k+N_p-1|k) \; \right]^T , \tag{14}$$

which is directly given by $[x(k)^T \; X_k^T]$ with the last entry suppressed.

With the previous discussion in mind, we proceed by using the qLPV model from Eq. (5) and the definitions from Eqs. (7), (9) and (10) to analytically provide a solution to the state evolution which is explicitly dependent on the scheduling sequence.

For the LTI case, the state evolution X_k, departing from $x(k)$, is expressed on a linear dependent basis w.r.t. $x(k)$ and to the sequence of control inputs U_k, as follows:

$$X_k = \mathcal{A} x(k) + \mathcal{B} U_k \, .$$

Analogously, for the qLPV case, since linearity is retained through the input-output channels (i.e. from u to x), the state evolution can be given in a quite similar fashion, but with parameter dependence on P_k appearing on the transition matrices, this is:

$$X_k \quad = \quad \mathcal{A}(P_k) x(k) + \mathcal{B}(P_k) U_k \, , \tag{15}$$

where the parameter dependent matrices are given by[1]:

$$\mathcal{A}(P_k) = \begin{bmatrix} A(\rho(k)) \\ A(\rho(k+1))A(\rho(k)) \\ \vdots \\ \left(\Pi_{i=0}^{N_p-1} A(\rho(k+i))\right) \end{bmatrix}, \quad \mathcal{B}(P_k) = \begin{bmatrix} B(\rho(k)) & \dots & \left(\Pi_{i=1}^{N_p-1} A(\rho(k+i))\right) B(\rho(k)) \\ 0 & \ddots & \left(\Pi_{i=2}^{N_p-1} A(\rho(k+i))\right) B(\rho(k+1)) \\ \vdots & \dots & \vdots \end{bmatrix}^T.$$

In order to compute matrices $\mathcal{A}(P_k)$ and $\mathcal{B}(P_k)$, some nonlinear operations should be performed. Anyhow, the procedure to compute them can be done completely outside the MPC optimization. In this form, the MPC receives, at each instant k, the following inputs: $x(k)$, $\mathcal{A}(P_k)$ and $\mathcal{B}(P_k)$ (as well as the steady-state target given by x_r and u_r); then, solving the optimization problem in Eq. (8), it results in U_k, from which the first entry $u(k|k)$ is applied to the plant. For such goal, the MPC requires to internally explicitly minimize the cost function $J(x,u,k)$ from Eq. (6), which can be written in the vector form as follows:

$$\begin{aligned} J(x,u,k) &= (X_k - X_r)^T \breve{Q} (X_k - X_r) + (U_k - U_r) \breve{R} (U_k - U_r) \\ &+ V(x(k+N_p|k)), \\ &= (\mathcal{A}(P_k)x(k) + \mathcal{B}(P_k)U_k - X_r)^T \\ &\quad \breve{Q} (\mathcal{A}(P_k)x(k) + \mathcal{B}(P_k)U_k - X_r) \\ &+ (U_k - U_r) \breve{R} (U_k - U_r) \\ &+ V(x(k+N_p|k)), \\ &= \frac{1}{2}\left(U_k^T H(P_k) U_k - U_k^T g(\cdot) + \kappa(\cdot)\right), \end{aligned} \tag{16}$$

where $H(P_k)$ is the Hessian of this cost function, $g(\cdot)$ its gradient and $\kappa(\cdot)$ an offset term. The notation $\breve{Q}$ and $\breve{R}$ denote the block-diagonal version of these matrices, i.e. $\text{diag}\{\overbrace{Q \dots Q}^{N_p \text{ times}}\}$ and $\text{diag}\{\overbrace{R \dots R}^{N_p \text{ times}}\}$, respectively, while $X_r = \begin{bmatrix} \overbrace{1 \dots 1}^{N_p \text{ times}} \end{bmatrix}^T x_r$ and $U_r = \begin{bmatrix} \overbrace{1 \dots 1}^{N_p \text{ times}} \end{bmatrix}^T u_r$. The Hessian, gradient and offset terms are analytically given by:

$$H(P_k) = 2\mathcal{B}(P_k)^T \breve{Q} \mathcal{B}(P_k) + 2\breve{R}, \tag{17}$$

$$\begin{aligned} g(\cdot) &= -\mathcal{B}(P_k)^T \breve{Q} \mathcal{A}(P_k) x(k) \\ &+ \mathcal{B}(P_k)^T \breve{Q} X_r + \breve{R} U_r, \end{aligned} \tag{18}$$

$$\begin{aligned} \kappa(\cdot) &= 2x(k)^T \mathcal{A}(P_k)^T \breve{Q} \mathcal{A}(P_k) x(k) \\ &- x(k)^T \mathcal{A}(P_k)^T \breve{Q} X_r + 2X_r^T \breve{Q} X_r \\ &+ 2U_r^T \breve{R} U_r + 2V(x(k+N_p|k)). \end{aligned} \tag{19}$$

[1] $\Pi(\cdot)$ denotes the left-side product operator.

3.1 Process constraints

The qLPV MPC problem solution proposed in this paper is formulated with respect to the scheduled state evolution equation, as gave Eq. (15), with $H(P_k)$, $g(\cdot)$ and $\kappa(\cdot)$ being as passed as inputs to the resulting MPC optimization. This means that the MPC optimization does not treat state evolution X_k as optimization variables, but the whole problem is formulated singularly in terms of U_k.

For this reason, the admissibility process constraints $u(k+i-1|k) \in \mathcal{U}$ and $x(k+i|k) \in \mathcal{X}$ are conversely written in the following fashion, w.r.t. $\mathcal{A}(\cdot)$, $\mathcal{B}(\cdot)$, U_k and $x(k)$, instead of $u(k+j)$ and $x(k+j)$:

$$U_k \in \breve{\mathcal{U}}, \tag{20}$$

$$(\mathcal{A}(P_k)x(k) + \mathcal{B}(P_k)U_k) \in \breve{\mathcal{X}}, \tag{21}$$

$$\mathcal{B}(P_k)U_k \in \breve{\mathcal{X}} \ominus \mathcal{A}(P_k)x(k). \tag{22}$$

The above formulations also facilitates the inclusion of slew rates on u, i.e. constraints on the control variation $\delta u(k+i) = u(k+i) - u(k+i-1)$, which can be done directly by adapting the vector-wise set $\breve{\mathcal{U}}$.

The terminal constraint $x(k+N_p|k) \in \mathbf{X}_f$ is stated in terms of U_k as:

$$\begin{bmatrix} 0_{n_x} & 0_{n_x} & \dots \mathbb{I}_{n_x} \end{bmatrix} (\mathcal{A}(P_k)x(k) + \mathcal{B}(P_k)U_k) \in \mathbf{X}_f,$$
$$\begin{bmatrix} 0_{n_x} & 0_{n_x} & \dots \mathbb{I}_{n_x} \end{bmatrix} \mathcal{B}(P_k)U_k \in X_f^{\ominus},$$

where

$$X_f^{\ominus} := \mathbf{X}_f \ominus \left(\begin{bmatrix} 0_{n_x} & 0_{n_x} & \dots \mathbb{I}_{n_x} \end{bmatrix} \mathcal{A}(P_k)x(k) \right).$$

Additional output constraints are also easily formulated. If the process has some outputs y_c (which are not necessarily equal to y, but could be) that must be hard constrained, i.e. $y_c \in \mathcal{Y}_c$. Then, assume that these outputs can be described as:

$$y_c(k) = C_c(\rho(k))x(k), \tag{23}$$

In what follows, we take

$$Y_{c_k} = \begin{bmatrix} y_c(k+1|k) & \dots & y_c(k+N_p-1)|k) \end{bmatrix}^T.$$

Then, the additional constraint is formulated as follows[2]:

$$\begin{aligned} C_c(P_k) &= \text{diag}\{ \; C_c(\rho(k+1) \quad \dots \quad C_c(\rho(k+N_p-1)) \quad 0 \; \}, \\ Y_{c_k} &= C_c(P_k)X_k, \\ Y_{c_k} &= C_c(P_k)\left(\mathcal{A}(P_k)x(k) + \mathcal{B}(P_k)U_k\right), \end{aligned}$$

thus

$$C_c(P_k)\mathcal{B}(P_k)U_k \in \breve{\mathcal{Y}}_c \ominus (C_c(P_k)\mathcal{A}(P_k)x(k)).$$

[2]Note that the last $y_c(k+N_p|k)$ is not constrained due to unavailability of $\rho(k+N_p)$ inside P_k. This issue is amendable by taking a longer scheduling prediction guess P_k, but out of the scope.

3.2 Reference tracking

Finally, before showing the proposed mechanism to guess P_k and solve the qLPV MPC problem, a comment must be made regarding reference tracking. The considered cost function $J(x, u, k)$ from Eq. (6) (or its vector form of Eq. (16)) is set in order to minimize the variations from the desired set-point target $p_r = (x_r , u_r)$.

The majority of processes that require reference tracking, require it regarding the controlled outputs and not the states. This is, to ensure that $y(k)$ tracks some steady-state value y_r. Since the controlled outputs in Eq. (5) are given by

$$y(k) = C(\rho(k))x(k),$$

we can find a linear (parameter varying) combination of the states x that, if tracked, ensures that $y(k) \rightarrow y_r$. We denote the output tracking target as $p_r^y = (y_r, u_r^y)$, which is known.

Then, following the lines of previous reference tracking frameworks[24–26], we use an offline reference optimization selector, which is set to find the set-point target p_r that abides to the constraints and ensures an output tracking of y_r. This nonlinear optimization procedure is as follows:

$$\begin{aligned}
\min_{x_r, u_r} \quad & \|C(\rho_r)x_r - y_r\|_Q^2 + \|u_r - u_r\|_R^2, \\
\text{s.t.} \quad & \begin{bmatrix} (\mathbb{I}_{n_x} - A(\rho_r)) & -B(\rho_r) \\ C(\rho_r) & 0_{nx \times n_u} \end{bmatrix} \begin{bmatrix} x_r \\ u_r \end{bmatrix} = \begin{bmatrix} 0_{n_x} \\ y_r \end{bmatrix}, \\
& \rho_r = f_\rho(x_r, u_r) \in \mathcal{P}, \\
& x_r \in \mathcal{X}, \\
& x_r \in \mathbf{X}_f, \\
& u_r \in \mathcal{U}.
\end{aligned}$$

This procedure ensures some steady-state $x_r = A(\rho_r)x_r + B(\rho_r)u_r)$ that abides to the states constraints and guarantees that the output tracking goal y_r is followed.

Note that this optimization procedure has a steady-state target point $p_r = (x_r , u_r)$ as output, and not the full state and input trajectories towards this target.

The state reference selection problem can be solved online, at each sampling instant, if the output reference goal y_r changes over time. By doing so, an additional computational complexity appears, which can be smoothed if the scheduling parameter guess P_k is used instead of solving the nonlinear optimization itself. A full discussion on periodically changing reference tracking for nonlinear MPC has been recently presented[27]. The focus of this paper is constant reference signals, either given in terms of states x_r or outputs y_r

It is important to notice that, in order for the method to hold, the state reference x_r must be contained inside the terminal set of the MPC problem from Eq. (8). This ensures that the stability and recursive feasibility guarantees (as verified in Section *Stability and Offline Preparations*) hold.

4. THE MHE-MPC MECHANISM

In the general qLPV embedding case of Eq. (5), the scheduling proxy $f_\rho(\cdot)$ is an arbitrary function of both state and input. For notation ease, we drop the control input dependency, using taking $\rho(k) = f_\rho(x(k))$. Anyhow, note that all that follows can be trivially extended to broader case.

The backbone idea of the method proposed in this paper follows the fashion of previous papers[19,22]: to iteratively refine the predictions/guesses of the scheduling sequence P_k based on the (adjusted) state predictions $X_k^\star$. The main novelty of this paper is how the refining and estimation of P_k is done: in the prior, the scheduling sequence is taken, as each iteration, directly as gives Eq. (13), i.e. as a nonlinear operator upon a vector, which can be computationally difficult to track, according to the kind of nonlinearity present on the scheduling map $f_\rho(\cdot)$; in contrast, in this paper, P_k is taken according to a linear time-varying operator on $X_k^\star$, this is: $P_k = f^{\text{MHE}}(P_{k-1}, \Theta_k, X_k^\star)$. This linear operator derives from a Moving-Horizon Estimation procedure, which proceeds by trying to match a fixed-size linear auto-regressive model for P_k, being Θ_k the model parameters computed by the MHE at a given sampling instant k and P_{k-1} the scheduling sequence guess at the previous instant.

A priori, the operation of this MHE has the computational complexity of a QP, which could be faster to evaluate than the Eqs. (13) with $\mathcal{A}(P_k)$ and $\mathcal{B}(P_k)$, depending on the amount of nonlinearities present in $f_\rho(\cdot)$. Moreover, the proposed MHE-MPC mechanism is able to provide convergence to the real scheduling sequence P_k faster than looping Eq. (13) to the MPC[22], as demonstrated through the experiment presented in Section *Benchmark Example*. The method follows:

Assumption 8. *The scheduling map $f_\rho(\cdot)$ is algebraic.*

Proposition 1. *The scheduling map $P_k = f_\rho(X_k^\star)$ can be approximated by the following regression:*

$$\begin{aligned}
\rho(k) &\approx \rho(k-1) + a_{0,0}x(k) + a_{0,1}x(k+1|k) \\
&+ \cdots + a_{j,N_p-1}x(k+N_p-1|k)\,, \\
&\vdots \\
\rho(k+N_p-1) &\approx \rho(k-N_p-2) + a_{N_p-1,0}x(k) \\
&+ a_{N_p-1,1}x(k+1|k) + \ldots \\
&\quad a_{N_p-1,N_p-1}x(k+N_p-1|k)\,,
\end{aligned} \tag{24}$$

which can be given in compact form by:

$$P_k \approx P_{k-1} + \underbrace{\begin{bmatrix} a_{0,0} & \cdots & a_{1,N_p-1} \\ \vdots & \ddots & \vdots \\ a_{N_p-1,1} & \cdots & a_{N_p-1,N_p-1} \end{bmatrix}}_{\Theta_k} X_k^\star\,,$$

being $f^{MHE}(P_{k-1}, \Theta_k, X_k^\star) = P_{k-1} + \Theta_k X_k^\star$ a fairly easy map to compute with respect to numerical burden.

Proof. Indeed, due to Assumption 8, it is quite reasonable that Assumption 1 holds: any algebraic function of form $\rho(k+1) = f_\rho(x(k+1))$ can be Taylor-expanded to achieve a linear dependency on x with sufficiently small error, i.e.

$$\begin{aligned}
\rho(k+1) - \rho(k) &= f_\rho(x(k+1) - f_\rho(x(k))\,, \\
\rho(k+1) &= \rho(k) + f_\rho(x(k+1)) - f_\rho(x(k))\,, \\
\rho(k+1) &\approx \rho(k) + \left.\frac{\partial f_\rho}{\partial x}\right|_{x(k+1)} (x(k+2) - x(k+1)) \\
&+ \left.\frac{\partial f_\rho}{\partial x}\right|_{x(k)} (x(k+1) - x(k))\,, \\
&\approx \rho(k-1) + a_2 x(k+2) \\
&+ a_1 x(k+1) + a_0 x(k)\,.
\end{aligned}$$

This concludes the proof. □

The proposed procedure uses a MHE mechanism to estimate these parameter values $a_{0,0}$ to a_{N_p-1,N_p-1}, at each sampling instant, concatenated as Θ_k, through the following QP:

$$
\begin{aligned}
\min_{\Theta_k} \quad & \sum_{j=0}^{N_p-1} \left(e^T(k+j) e(k+j) \right) \qquad (25)\\
\text{s.t.} \quad & e(k+j) = \overbrace{x(k+j|k)}^{\text{Data from } X_k^\star} - \hat{x}(k+j), \forall\, j \in \mathbb{N}_{[0,N_p-1]} \\
& \hat{x}(k+j) = A(\rho(k+j-1))x(k+j-1|k) \\
& \quad + B(\rho(k+j-1)) \overbrace{u(k+j-1)}^{\text{Data from } U_k} \forall\, j \in \mathbb{N}_{[0,N_p-1]}, \\
& \rho(k+j) = \overbrace{\rho(k+j-1)}^{\text{Data from } P_{k-1}} + a_{j,0}x(k) + \ldots \\
& \quad + a_{j,N_p-1}x(k+N_p-1|k) \forall\, j \in \mathbb{N}_{[0,N_p-1]} \\
& \rho(k+j) \in \mathcal{P}, \ \forall\, j \in \mathbb{N}_{[0,N_p-1]}.
\end{aligned}
$$

black

Essentially, this MHE scheme operates in order to find a parameter matrix Θ_k that makes the linear auto-regressive equation $P_k = P_{k-1} + \Theta_k X_k^\star$ yield the *best match* between the state evolution sequence $X_k^\star$ and the qLPV model. Notice that the MHE algorithm only needs a few data from the previous step to find the parameter matrix Θ_k, being these the state predictions $X_k^\star$, the scheduling sequence P_{k-1} and the input vector U_k.

Figure 1 syntheses the proposed algorithm, which relies in a coordination between the MHE and the MPC optimization procedures. We must note that the MHE loop should operate until P_k converges to the actual value for the scheduling sequence, or until a certain *stop criterion*/heuristic threshold for the number of iterations is reached.

The proposed scheme is also detailed through the Algorithm below. Its application departs with an initial state evolution sequence X_0, that can be simply taken as constant/frozen evolution for the states and two initial scheduling sequences P_0 and P_{-1}, which can be simply taken as if $\rho(k)$ remained frozen along the whole prediction horizon, i.e. $P_k = \rho(k)\mathbb{I}_{1,N_p}$. The Algorithm also departs with a known terminal set condition $\mathbf{X}_f$ and a known target reference goal p_r. The Hessian, gradient and offset of J_k are taken as give Eqs. (17)-(19), respectively.

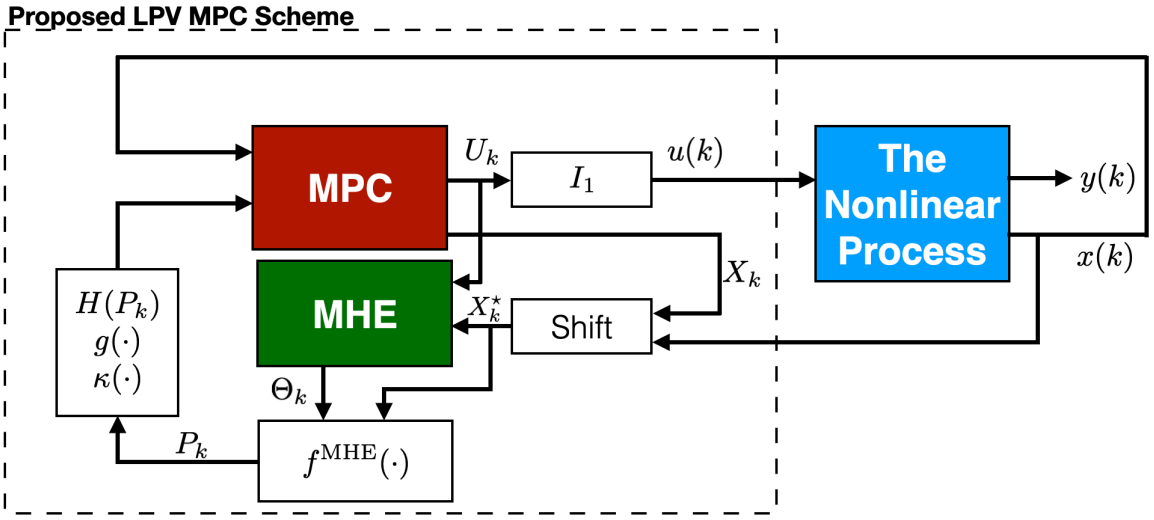

Figure 1. Proposed MHE-MPC Scheme using a qLPV Model.

Algorithm 1 Proposed qLPV MPC

Initialize: $x(0) = x_0, \rho(0) = \rho_0, k = 0$.
Require: $\check{Q}, \check{R}, N_p, p_r$.
Require: P_0, P_{-1}, X_0, U_0.
Loop:

- Step (1): **Loop until convergence of** P_k:
  - (i) Shift and update $X_k \implies X_k^\star$;
  - (ii) Based on $X_k^\star$, P_{k-1} and U_k, solve the MHE optimization, from Eq. (25), which yields Θ_k;
  - (iii) Compute $P_k = f^{\text{MHE}}(P_{k-1}, \Theta_k, X_k^\star)$ according to Eq. (24);
  - (iv) Compute Hessian $H(P_k)$, gradient $g(\cdot)$ and offset $\kappa(\cdot)$;
  - (v) Solve the MPC optimization in Eq. (8), which yields U_k and X_k.
- Step (2): Take $u(k) = I_1 U_k$ (first entry of the vector);
- Step (3): Apply this local control policy to the nonlinear process.
- Step (4): Increment k, i.e. $k \leftarrow k + 1$.

end

4.1 Convergence property

In order to demonstrate the convergence of the proposed method (as in its implementation form given in Algorithm 1), we will proceed by verifying a well-known results for Newton based SQPs from the literature [28–30]. This is the same path followed in a previous paper [22], which invokes established results to demonstrate that, under certain conditions, the MHE-MPC mechanism that is solved at each iteration is equivalent to a quadratic sub-problem used in standard Newton SQP. Therefore, a local convergence property is readily found.

Proposition 2. *A quadratic sub-problem program of SQP algorithms is derived by a second-order approximation of the SQP optimization cost and a linearization of its constraints.*

Proof. Found in [28]. □

For illustration purposes regarding this matter, we consider the following generic NP (as given in Definition

1):

$$\begin{aligned} \min_{x_c} \quad & f_c(x_c)\,, \\ \text{s.t.} \quad & h_j(x_c) \;=\; 0\,, \\ & g_i(x_c) \;\leq\; 0\,. \end{aligned} \tag{26}$$

This problem can be given as a quadratic sub-problem directly, as follows:

$$\begin{aligned} \min_{\breve{x}_c} \quad & \left(\breve{x}_c^T \left(H_{f_c}(x_c)|_{x_c=\overline{x}_c}\right)\breve{x}_c + \left(\nabla f_c(x_c)|_{x_c=\overline{x}_c}\right)^T \breve{x}_c\right)\,, \\ \text{s.t.} \quad & \left(\nabla h_j(x_c)|_{x_c=\overline{x}_c}\right)\breve{x}_c + \left(\nabla h_j(x_c)|_{x_c=\overline{x}_c}\right) \;=\; 0\,, \\ & \left(\nabla g_i(x_c)|_{x_c=\overline{x}_c}\right)\breve{x}_c + \left(\nabla g_i(x_c)|_{x_c=\overline{x}_c}\right) \;\leq\; 0\,, \end{aligned} \tag{27}$$

where $H_{f_c}(x_c)$ denotes the Hessian of the optimization cost $f_c(x_c)$ and $\nabla h_j(x_c)$ and $\nabla g_i(x_c)$ denote divergent operators. We note that this sub-problem is evaluated at a given solution estimate $\overline{x}_c$ (at some given iteration), for which $\breve{x}_c \;=\; x_c - \overline{x}_c$.

Regarding the proposed MHE-MPC mechanism, we can easily show that if either simple Jacobian linearization or Linear Differential Inclusion are used to find a qLPV model (as in Eq. (5)) for the nonlinear system (as in Eq. (1)), then, the proposed mechanism iterates in equivalence to a Newton SQP sub-problem. Notice how such sub-problem in Eq. (27) is identical to the MHE-MPC optimization given through the consecutive iterations of the MHE (Eq. (25)) and the MPC (Eq. (8)). The terminal constraint in the MPC optimization adds no convergence trade-off.

Thence, it follows that if local convergence of the equivalent Newton SQP can be established, the proposed MHE-MPC also yields convergence. The sufficient conditions for local convergence of a Newton SQP sub-problem at $x_c \;=\; \overline{x}_c$, as given by prior references[29–32], are that (*i*) the problem is set simply with equality constraints (not the MPC case) or that (*ii*) the subset of active inequality constraints are known before the optimization solution. The second condition is also not true for general MPC paradigms. However, one can iterate the sub-problem until convergence is found at another point $x_c \;=\; \overline{\overline{x}}_c$, as previously discussed[22,33,34].

In practice, the proposed MHE-MPC will not be set to freely iterate until the convergence of P_k. This is not desirable because the number of iterations needed for convergence may require more time than the available sampling period. Therefore, a stop criterion is added to the mechanism, so that iterations stop at a given threshold. A warm-start is also included by shifting the result regarding X_k and P_k from one sampling k as the initial guess for the optimization at $k+1$, which ensures that the proposed algorithm reaches convergence after a few discrete-time samples. We note that convergence of Newton SQP sub-problems with warm-start have been assessed[33,34] and shown to be practicable for real-time schemes.

5. STABILITY AND OFFLINE PREPARATIONS

In this Section, we offer a Theorem to construct the terminal ingredients of the MPC algorithm: (a) the terminal set within which $x(k+N_p|k)$ is bounded to, and (b) the terminal offset cost $V(x(k+N_p|k))$ minimized by the MPC.

Since the establishment of terminal ingredients toolkit[2,35] as the key way to ensure stability and recursive feasibility of state-feedback predictive control loops, MPC grew on both theory and industrial practice.

The usual approach with terminal ingredients resides in some ensuring that conditions are met by (a) the terminal set $\mathbf{X}_f$ and (b) the terminal cost $V(x(k+N_p|k))$ with respect to a nominal state-feedback controller

$u(k) = K^{\mathrm{n}}x(k)$, which is usually the unconstrained solution of the MPC problem. For the tracking case, the nominal feedback is given by $u(k) = K^{\mathrm{n}}(x(k) - x_r)$ (and so is the terminal constraint $\left(x(k+N_p|k) - x_r\right) \in \mathbf{X}_f$ and the terminal cost $V(x(k+N_p|k) - x_r)$). Accordingly, we develop a sufficient stability condition for the proposed MHE-MPC mechanism in order to verify these conditions.

Firstly, we consider that there exists a parameter-dependent nominal state-feedback gain $K^{\mathrm{n}} : \mathbb{R}^{n_p} \rightarrow \mathbb{R}^{n_x \times n_u}$. For demonstration simplicity and notation lightness, we will proceed with x_r null[3].

This nominal controller is purely fictional, used to demonstrate stability and recursive feasibility properties of the proposed MHE-MPC mechanism. Anyhow, it stands for the infinite-horizon LPV Linear Quadratic Regulator (LQR) solution, which verifies $K^{\mathrm{n}}(\rho) = \arg\min_{K \in \mathbb{R}^{n_x \times n_u}} \left(\sum_{i=1}^{+\infty} \|x(k+i|k)\|_Q^2 + \|Kx(k+i-1)\|_R^2\right)$ and the admissibility of $x(k+i)$ and $u(k+i-1)$.

Of course, there is a complexity barrier to solve this problem, because the states have parametric nonlinearities that impact their trajectories (the qLPV scheduling parameters). Therefore, we determine this nominal feedback gain together with the terminal ingredients, which are also taken as parameter-dependent on ρ. We consider, for regularity, an ellipsoidal set as the terminal constraint, which is given by:

$$\mathbf{X}_f = \left\{x \,|\, x^T P(\rho) x \leq \alpha_P\right\}. \tag{28}$$

This ellipsoid is centered at the origin and has a radius of α_P. Furthermore, this terminal set is a sub-level set of terminal cost $V(\cdot)$, which is taken as a Lyapunov function as follows:

$$V(x, \rho) = x^T P(\rho) x. \tag{29}$$

This parameter-dependent nominal feedback gain $K^{\mathrm{n}}(\rho))$ and the parameter dependent terminal ingredients verbalized through the symmetric parameter dependent Lyapunov matrix $P(\rho)$ are so that the following input-to-state stability Theorem is guaranteed.

Theorem 1. *Input-to-State Stable MPC*[2,22,35,36]
Let Assumptions 4 and 7 hold. Assume that a nominal control law $u = K^n(\rho)x$ exists. Consider that the MPC is in the framework of the optimization problem in Eq. (8), with 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 are hold $\forall \rho \in \mathcal{P}$:

- *(C1) The origin lies in the interior of $\mathbf{X}_f(\rho)$;*
- *(C2) Any consecutive state to x, given by $(A(\rho) + B(\rho)K^n(\rho))\,x$ lies within $\mathbf{X}_f(\rho)$ (i.e. this is an invariant set);*
- *(C3) The discrete algebraic Ricatti equation is verified within this invariant set, this is, $\forall\, x \in \mathbf{X}_f(\rho(k))$:*

$$\begin{aligned} V\left(\left(A(\rho(k)) + B(\rho(k))K^n(\rho(k))\right)x, \rho(k+1)\right) - V\left(x, \rho(k)\right) \\ \leq -x^T Q x - x^T \left(K^n(\rho(k))\right)^T R K^n(\rho(k)) x. \end{aligned}$$

- *(C4) The image of the nominal feedback always lies within the admissible control input domain: $K^n(\rho(k))x \in \mathcal{U}$.*
- *(C5) The terminal set $\mathbf{X}_f(\rho)$ is a subset of the admissible state domain $\mathcal{X}$.*

[3]The tracking equivalency is easily done with $\frac{dx_r}{dt} = 0$ and by computing the qLPV model with the states dynamics are given with respect to $x_{tracking}(k) = x(k) - x_r$.

Assuming that the initial solution of the MPC problem $U^\star$, computed with respect to the initial state $x(0)$, is feasible, the MPC algorithm is indeed recursively feasible, asymptotically stabilizing the state origin.

Proof. Provided in Appendix *Proof of Theorem 1*. □

In order to find some nominal state-feedback gain $K^{\mathrm{n}}(\rho)$, some terminal set $\mathbf{X}_f$ and some terminal offset cost $V(\cdot)$, an offline LMI problem is proposed in the sequel. This LMI problem is such that a $P(\rho)$ positive definite parameter-dependent matrix is found to ensure that the conditions of Theorem 1 are satisfied. Due to condition (C3), the LMI is solved over a sufficiently dense grid over ρ, consider its admissibility domain $\mathcal{P}$. We note that (C3) is a time-variant condition, which depends explicitly on $\rho(k)$ and $\rho(k+1)$ due to the nature of the parameter dependent $V(\cdot)$.

This LMI problem is provided through the following Theorem, which aims to find the largest terminal set $\mathbf{X}_f$ that is invariant under the nominal control policy $u(k) = K^{\mathrm{n}}(\rho(k))x(k)$ for all k, while remaining admissible, i.e. $K^{\mathrm{n}}(\rho)x \in \mathcal{U}$, $\forall x \in \mathcal{X}$ and $\rho \in \mathcal{P}$. Note that the largest ellipsoidal set as in the form of Eq. (28) is posed through the maximization of α_P.

Theorem 2. *Terminal Ingredients*[22,37]
The conditions (C1)-(C5) of Theorem 1 are satisfied if there exist a symmetric parameter-dependent positive definite matrix $P(\rho) : \mathbb{R}^{n_p} \to \mathbb{R}^{n_x \times n_x}$, a parameter-dependent rectangular matrix $W(\rho) : \mathbb{R}^{n_p} \to \mathbb{R}^{n_u \times n_x}$ and a scalar $0 < \hat{\alpha}_P \in \mathbb{R}$ such that $Y(\rho) = (P(\rho))^{-1} > 0$, $W(\rho) = K^n(\rho)Y(\rho)$ and that the following LMIs hold for all $\rho \in \mathcal{P}$ and $\partial\rho \in \partial\mathcal{P}$, while minimizing $\hat{\alpha}_P$:

$$\left[\begin{array}{cc|cc} Y(\rho) & \star & \star & \star \\ (A(\rho)Y(\rho) + B(\rho)W(\rho)) & Y(\rho+\partial\rho) & \star & \star \\ \hline Y(\rho) & 0 & Q^{-1} & \star \\ W(\rho) & 0 & 0 & R^{-1} \end{array}\right] \geq 0, \tag{30}$$

$$\left[\begin{array}{c|c} \hat{\alpha}_P \overline{u}_i^2 & I_i W(\rho) \\ \hline \star & Y(\rho) \end{array}\right] \geq 0,\ i \in \mathbb{N}_{[1,n_u]}, \tag{31}$$

$$\left[\begin{array}{c|c} \hat{\alpha}_P \overline{x}_j^2 & I_j Y(\rho) \\ \hline I_j^T & \mathbb{I}_{n_x} \end{array}\right] \geq 0,\ j \in \mathbb{N}_{[1,n_x]}, \tag{32}$$

where I_j denotes the j-th row of the identity matrix $\mathbb{I}$. In LMI (31), it is given w.r.t. to an identity $\mathbb{I}_{n_u}$, while in LMI (32), it is given w.r.t. to an identity $\mathbb{I}_{n_x}$.

Proof. Refer to[37]. □

We must note that the above proof demonstrates that the solution of the LMIs presented in Theorem 2 ensure a positive definite parameter dependent matrix $P(\rho)$ which can be used to compute the MPC terminal ingredients $V(\cdot)$ and $\mathbf{X}_f$ such that input-to-state stability of the closed-loop in guaranteed, verifying the conditions of Theorem 1. Furthermore, when the MPC is designed with these terminal ingredients, for whichever initial condition $x(0) \in \mathbf{X}_f$ it starts with, it remains recursively feasible for all consecutive discrete time instants $k > 0$.

Anyhow, Theorem 2 provides infinite-dimensional LMIs, since they should hold for all $\rho \in \mathcal{P}$ and for all $\partial\rho \in \partial\mathcal{P}$. To address this issue, one can handle the LMIs considering an sufficiently dense grid[38] of $\mathbb{R}^{n_p \times n_p}$ points in $\mathcal{P} \times \partial\mathcal{P}$, for which the LMIs must be enforced. This solves the infinite dimension of the problem, which is converted into an n_g-dimensional LMI problem, being n_g the number of grid points. For this solution

Table 1. Model Parameters and Constraints

Parameter	Value/Set	Parameter	Value/Set
q	$100\,\mathrm{L\,min^{-1}}$	C_{Af}	$1\,\mathrm{mol\,L^{-1}}$
k_0	$7.2\times10^{10}\,\mathrm{min^{-1}}$	E/R	8750 K
H_Δ	$-5\times10^{5}\,\mathrm{cal\,mol^{-1}}$	ρC_p	$239\,\mathrm{cal\,L^{-1}\,K^{-1}}$
W	$7\times5\,10^{4}\,\mathrm{cal\,min^{-1}\,K^{-1}}$	V	100 L
T_f	350 K	–	–
C_A	$\in [0.03, 0.12]\,\mathrm{mol\,l^{-1}}$	T	$\in [440, 460]$ K
T_c	$\in [200, 380]$ K	–	–

to be practically implementable, continuity on matrices $A(\rho)$ and $B(\rho)$ should be verified. We must also note that parameter-dependency on ρ may be dropped if the system is quadratically stabilizable, which is a conservative assumption.

6. BENCHMARK EXAMPLE

In this Section, we pursue the application of the proposed MHE-MPC mechanism with terminal ingredients found through Theorem 2. For such, we consider the application of our control method upon a benchmark system, detailed in the sequel.

6.1 Continuously-stirred tank reactor

Consider the model of a Continuously-Stirred Tank Reactor (CSTR) process, which consists of an irreversible, exothermic reaction, $A \rightarrow B$, in a constant volume reactor cooled by a single coolant stream which can be modeled by the following equations:

$$\begin{aligned}\dot{C}_A &= \frac{q}{V}(C_{Af} - C_A) - k_0 e^{\frac{-E}{RT}} C_A \\ \dot{T} &= \frac{q}{V}(T_f - T) - \frac{H_\Delta}{\rho C_\rho} k_0 e^{\frac{-E}{RT}} C_A + \frac{W}{V\rho C_\rho}(T_c - T)\end{aligned} \tag{33}$$

where C_A is the concentration of A in the reactor, T is the temperature in the reactor, and T_c is the coolant temperature.

In this process, $u = T_c$ is a control input, whereas C_A and T are measurable process variables. The considered model parameters and process constraints are reported in Table 1.

6.2 qLPV Embedding

Considering $x = (C_A, T)$ as state variables, we obtain the following qLPV realization of the CSTR system from Eq. (33):

$$\dot{x} \quad = \quad A(\rho)x + Bu\,, \tag{34}$$

where $\rho = f(x)$ denotes two scheduling variables, given as linear functions of each state, i.e. $\rho_1 = f_1(x_1)$ and $\rho_2 = f_2(x_2)$. This continuous-time model is Euler-discretized using a sampling period of $T_s = 30$ ms.

6.3 Control goal and tuning

Considering an arbitrary initial condition x_0 given within the state admissibility set $\mathcal{X}$, the proposed controller is set to steer the state trajectories to a known state reference x_r. For such, we use identity weights in the MPC cost $J(x, u, k)$. Complementary, we use a prediction horizon of $N_p = 8$ steps. This prediction horizon was chosen in accordance with prior literature using the same nonlinear CSTR benchmark[39].

The MHE scheme, which is used to estimate the future scheduling behaviour P_k at each sampling instant k, is set to operate with a threshold loop barrier of 3 loops, which is a verified sufficient bound to induce convergence (refer to the discussion in Section *Convergence Property*).

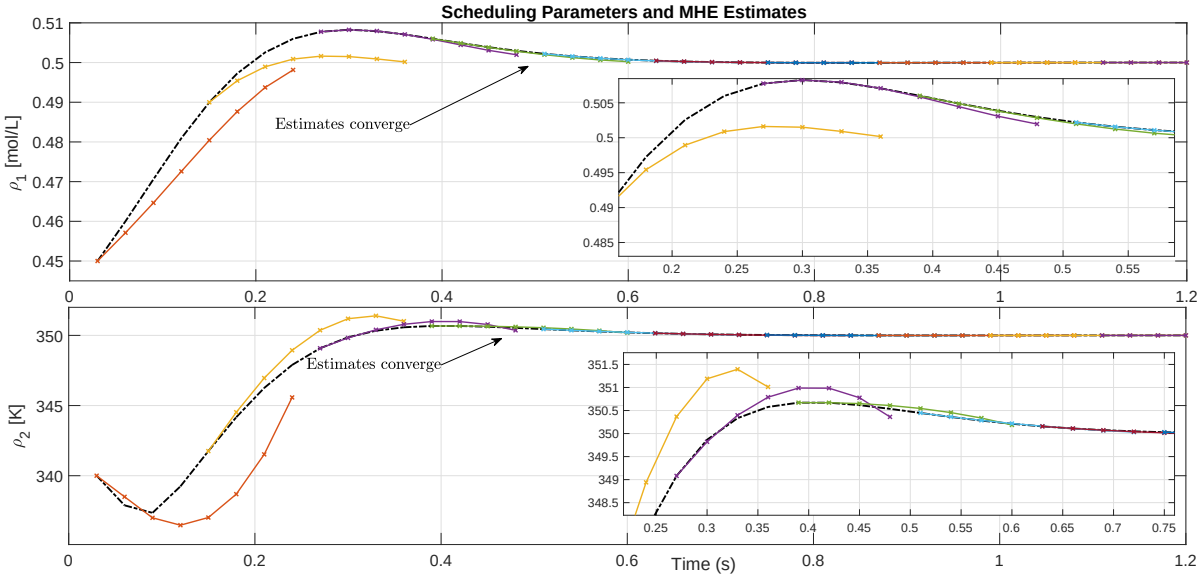

Figure 2. MHE: scheduling trajectory estimates.

6.4 Simulation results

Considering a realistic nonlinear CSTR model, we obtain simulation results to demonstrate the effectiveness of the proposed control scheme. The following results were obtained in a 2.4 GHz, 8 GB RAM Macintosh computer, using Matlab, yalmip and Gurobi solver.

First, we show how the MHE operation is able to accurately predict the behaviour of the scheduling trajectory, as depicts Figure 2. At each instant k, the MHE provides estimation for P_k, which is composed of the following N_p entries of the scheduling variables. In this Figure, we observe the real scheduling variables ρ_1 and ρ_2 (dot-dash black line) and the estimates P_k provided at different samples (coloured x-marked lines). Within some samples, we can see that the predicted trajectory converges to the real one, which confirms the effectiveness of the MHE operation.

Based on the scheduling behaviours predicted by the MHE loop, the predictive controller determines the control input (Figure 4) in order to drive the system states from x_0 to the reference goal x_r. The corresponding state trajectories are depicted in Figure 3, which also shows the state admissibility set $\mathcal{X}$ and the terminal set constraint $\mathbf{X}_f$ (a parameter-dependent ellipsoid generated via Theorem 2). As one can see, the behaviour of the process variables is a smooth trajectory towards x_r.

Finally, we demonstrate the dissipating properties of the proposed control scheme. In Figure 5, we show the evolution of MPC stage cost J over time. As expected, J decays and converges to the origin, which verifies the dissipation properties required by Theorem 1.

7. CONCLUSIONS

In this paper, a new method for the fast, real-time implementation of Nonlinear Model Predictive Control is proposed. The method provides a near-optimal, approximated solution, which is found through the on-line operation of sequential Quadratic Programming Problems. The main necessary argument to develop the method is that the nonlinear process should be described by quasi-Linear Parameter Varying model, for which the embedding is ensured through a scheduling proxy. Then, the online operations resides in the consecutive operation of the MPC program together with a Moving-Horizon Estimation scheme, which is used to match

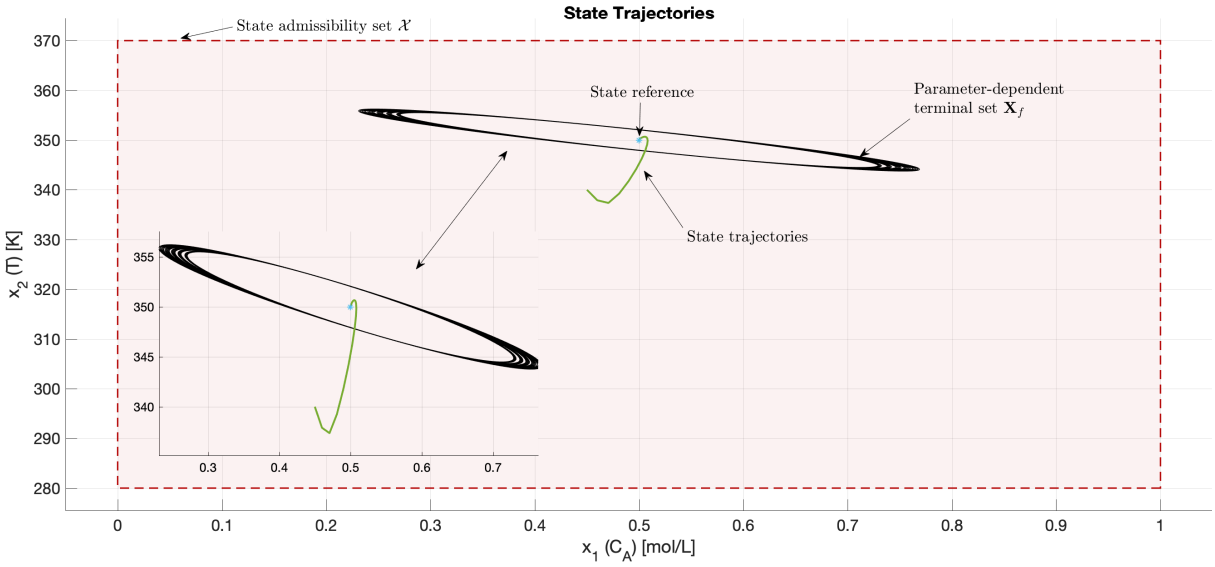

Figure 3. State trajectories, state admissibility set, terminal set.

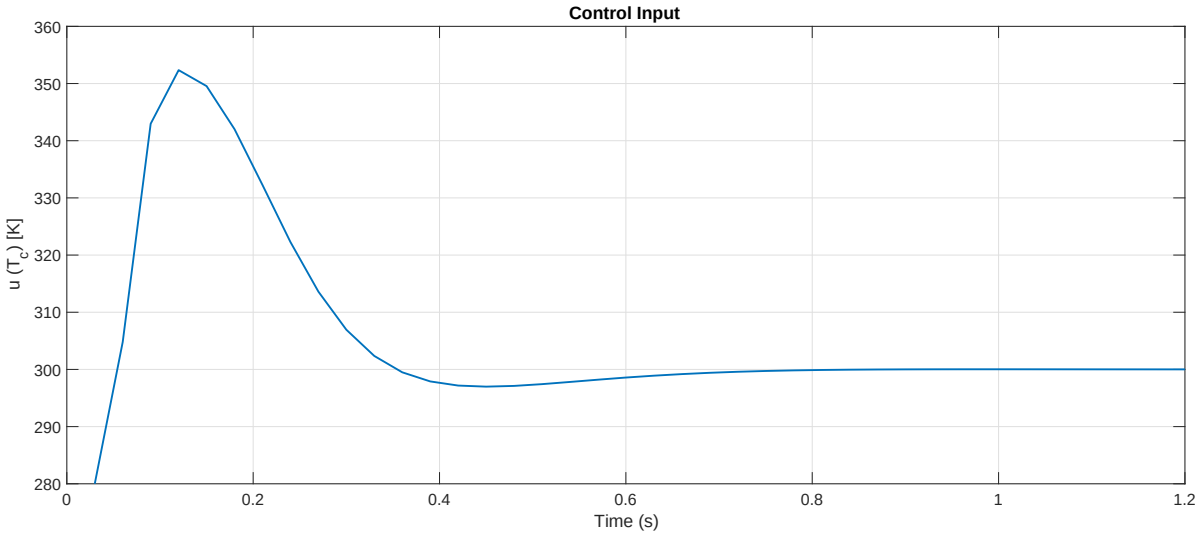

Figure 4. MPC: control input.

the future values of the scheduling proxy along the prediction horizon, which are unknown. Input-to-state stability and recursive feasibility properties of the algorithm are ensured by parameter-dependent terminal ingredients, which are computed offline. Using a benchmark example, the method is tested. We highlight that it proves itself more effective for stronger nonlinearities in the qLPV scheduling proxy, for which the MHE scheme operates faster than the application of the scheduling proxy upon each entry of the future state variables, as in many other techniques. For future works, the Authors plan on assessing the issue of periodically-changing (possibly unreachable) output reference signals.

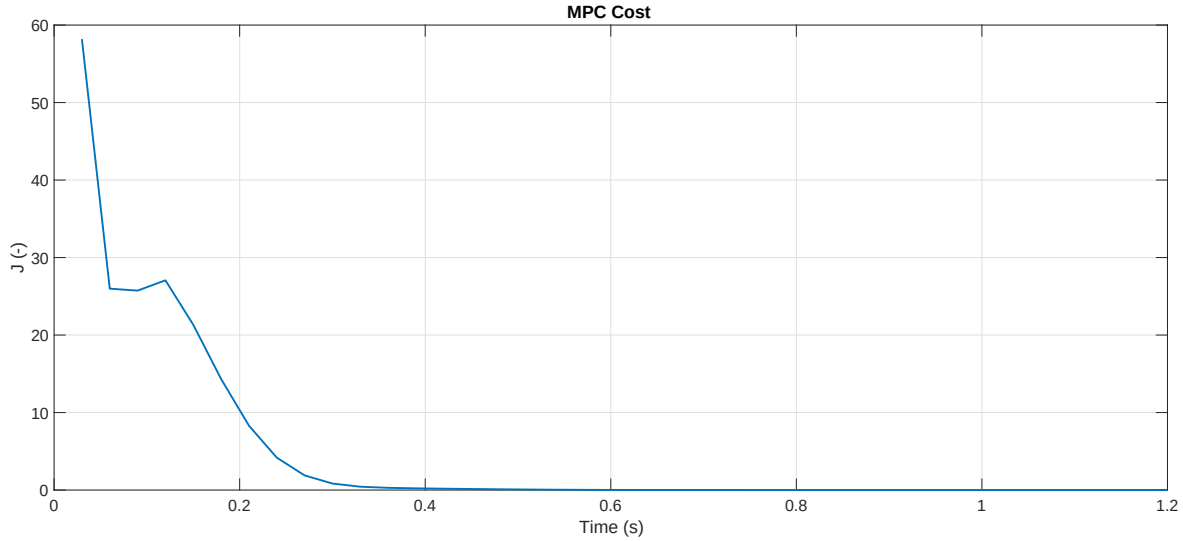

Figure 5. MPC: stage cost.

DECLARATIONS

Authors' contributions

All authors contributed equally.

Availability of data and materials

Data will be made available upon e-mail request to the corresponding author.

Financial support and sponsorship

M. M. Morato is partially supported by *CNPq* project 304032/2019 – 0. V. Stojanovic thanks the Serbian Ministry of Education, Science and Technological Development for support (grant No. 451-03-9/2021-14/200108).

Conflict of interest

Both Authors declare no potential conflict of interests.

Ethical approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

REFERENCES

1. Camacho EF, Bordons C. *Model predictive control*. Springer Science & Business Media, 2013.
2. Mayne DQ, Rawlings JB, Rao CV, Scokaert POM. Constrained model predictive control: Stability and optimality. *Automatica*, 2000;36:789–814.
3. Allgöwer F, Zheng A. *Nonlinear model predictive control*, volume 26. Birkhäuser, 2012.
4. Camacho EF, Bordons C. Nonlinear model predictive control: An introductory review. In *Assessment and future directions of nonlinear model predictive control*. Springer; 2007. p. 1–16.
5. Gros S, Zanon M, Quirynen R, Bemporad A, Diehl M. From linear to nonlinear MPC: bridging the gap via the real-time iteration. *International Journal of Control* 2020; 93:62–80 .

6. Quirynen R, Vukov M, Zanon M, Diehl M. Autogenerating microsecond solvers for nonlinear MPC: a tutorial using ACADO integrators. *Optimal Control Applications and Methods* 2015;36:685–704.
7. Englert T, Völz A, Mesmer F, Rhein S, Graichen K. A software framework for embedded nonlinear model predictive control using a gradient-based augmented lagrangian approach (GRAMPC). *Optimization and Engineering* 2019; 20:769–809,.
8. Ohyama S, Date H. Parallelized nonlinear model predictive control on GPU. In *11th Asian Control Conference*, pages 1620–1625. IEEE, 2017.
9. Rathai KMM, Sename O, Alamir M. GPU-based parameterized nmpc scheme for control of half car vehicle with semi-active suspension system. *IEEE Control Systems Letters* 2019; 3:631–6.
10. Hoffmann C, Werner H. A survey of linear parameter-varying control applications validated by experiments or high- fidelity simulations. *IEEE Transactions on Control Systems Technology* 2014;23:416–33 .
11. Morato MM, Normey-Rico JE, Sename O. Model predictive control design for linear parameter varying systems: A survey. *Annual Reviews in Control* 2020;49:64–80.
12. Boyd S, El Ghaoui L, Feron E, Balakrishnan V. *Linear matrix inequalities in system and control theory*, volume 15. Siam, 1994.
13. Abbas HS, Toth R, Petreczky M, Meskin N, Mohammadpour J. Embedding of nonlinear systems in a linear parameter-varying representation. *IFAC Proceedings Volumes* 2014; 47:6907–13.
14. Kunz K, Huck SM, Summers TH. Fast model predictive control of miniature helicopters. In *2013 European Control Conference (ECC)*, pages 1377–1382. IEEE, 2013.
15. Pablo SG Cisneros and Herbert Werner. Wide range stabilization of a pendubot using quasi-LPV predictive control. *IFAC-PapersOnLine* 2019;52:164–9.
16. Alcalá E, Puig V, Quevedo J. LPV-MPC control for autonomous vehicles. *IFAC-PapersOnLine* 2019;52:106–13.
17. Mate S, Kodamana H, Bhartiya S, Nataraj PSV. A stabilizing sub-optimal model predictive control for quasi-linear parameter varying systems. *IEEE Control Systems Letters*, 2019.
18. Morato MM, Normey-Rico JE, Sename O. Novel qLPV MPC design with least-squares scheduling prediction. *IFAC-PapersOnLine* 2019;52:158–63.
19. Morato MM, Normey-Rico JE, Sename O. Sub-optimal recursively feasible linear parameter-varying predictive algorithm for semi-active suspension control. *IET Control Theory & Applications* 2020;14:2764–75.
20. Cisneros PSG, Voss S, Werner H. Efficient nonlinear model predictive control via quasi-LPV representation. In *IEEE Conference on Decision and Control*, IEEE, 2016. p. 3216–21.
21. Cisneros PG, Werner H. Fast nonlinear MPC for reference tracking subject to nonlinear constraints via quasi-LPV representations. *IFAC-PapersOnLine* 2017; 50:11601–6 .
22. Cisneros PS, Werner H. Nonlinear model predictive control for models in quasi-linear parameter varying form. *International Journal of Robust and Nonlinear Control*, 2020.
23. Jungers M, Caun RP, Oliveira RCLF, Peres PLD. Model predictive control for linear parameter varying systems using path-dependent lyapunov functions. *IFAC Proceedings Volumes* 2009;42:97–102.
24. Limon D, Ferramosca A, Alvarado I, Alamo T, Camacho EF. MPC for tracking of constrained nonlinear systems. In *Nonlinear model predictive control*, Springer, 2009. p. 315–23.
25. Limon D, Ferramosca A, Alvarado I, Alamo T. Nonlinear MPC for tracking piece-wise constant reference signals. *IEEE Transactions on Automatic Control* 2018;63:3735–50.
26. Morari M, Maeder U. Nonlinear offset-free model predictive control. *Automatica* 2012;48:2059–67.
27. Köhler J, Müller MA, Allgöwer F. A nonlinear tracking model predictive control scheme for dynamic target signals. *Automatica* 2020;118:109030.
28. Qi L. Superlinearly convergent approximate newton methods for lc^1 optimization problems. *Mathematical programming* 1994 64:277–94.
29. Wei Z, Liu L, Yao S. The superlinear convergence of a new quasi-newton-sqp method for constrained optimization. *Applied mathematics and computation* 2008;196:791–801.
30. Izmailov AF, Solodov MV. On attraction of linearly constrained lagrangian methods and of stabilized and quasi-newton sqp methods to critical multipliers. *Mathematical programming* 2011;126:231–57.
31. Boggs PT, Tolle JW, Kearsley AJ. On the convergence of a trust region SQP algorithm for nonlinearly constrained optimization problems. In *System Modelling and Optimization,* Springer, 1996. p. 3–12 .
32. Boggs PT, Tolle JW. Sequential quadratic programming for large-scale nonlinear optimization. *Journal of computational and applied mathematics* 2000;124:123-137.
33. Diehl M, Bock HG, Schlöder JP. A real-time iteration scheme for nonlinear optimization in optimal feedback control. *SIAM Journal on control and optimization* 2005;43:1714–36.
34. Houska B, Ferreau HJ, Diehl M. An auto-generated real-time iteration algorithm for nonlinear MPC in the microsecond range. *Automatica*,2011;47:2279–85.
35. Michalska H, Mayne DQ. Robust receding horizon control of constrained nonlinear systems. *IEEE transactions on automatic control* 1993;38:1623–33.

36. Duan GR, Yu HH. *LMIs in control systems: analysis, design and applications*. CRC press, 2013.
37. Morato MM, Normey-Rico J, Sename O. Short-sighted robust lpv model predictive control: Application to semi-active suspension systems. In *European Control Conference 2021 (ECC21)*, pages 1–7, 2021.
38. Wu F. A generalized LPV system analysis and control synthesis framework. *International Journal of Control* 2001;74:745–59.
39. Chen H, Kremling A, Allgöwer F. Nonlinear predictive control of a benchmark CSTR. In *Proceedings of 3rd European control conference*, pages 3247–3252, 1995.

PROOF OF THEOREM 1

Consider an initial state condition $x(k_0)$ with a scheduling parameter $\rho(k_0)$ (transition map). Assume that this initial condition generates a feasible optimal control sequence

$$U^{\star}_{k_0} = \left[\begin{array}{cccc} u^{\star}(k_0|k_0) & u^{\star}(k_0+1|k_0) & \dots & u^{\star}(k_0+N_p-1|k_0) \end{array} \right]^T .$$

The next feasible sequence, which might be possibly sub-optimal, at instant k_0+1, is denoted:

$$U^{\star}_{k_0+1} = \left[\begin{array}{ccc} u^{\star}(k_0+1|k_0+1) & \dots & u^{\star}(k_0+N_p|k_0+1) \end{array} \right]^T .$$

Notice that the last entry of this second sequence $u(k_0+N_p|k_0+1)$ is equal to the feedback of the terminal state from the first sequence, this is:

$$u^{\star}(k_0+N_p|k_0+1) = K^{\mathrm{n}}(\rho)x(k_0+N_p|k_0+1) .$$

Then, let us[4] evaluate the evolution the MPC cost function $J(\cdot)$ and its decay between time instants k_0 and k_0+1:

$$\begin{aligned} J(k_0) &= \sum_{i=1}^{N_p} \left(\|x(k_0+i)^T Q x(k_0+i) \right) \\ &+ \sum_{i=0}^{N_p-1} \left(\|u(k_0)^T R u(k_0) \right) + V(x(k_0+N_p)) , \\ J(k_0+1) &= \sum_{i=1}^{N_p} \left(\|x(k_0+1+i)^T Q x(k_0+1+i \right) \\ &+ \sum_{i=0}^{N_p-1} \left(\|u(k_0+1)^T R u(k_0+1) \right) \\ &+ V(x(k_0+1+N_p)) . \end{aligned}$$

Assuming that $u^{\star}(k_0|k_0)$ was applied to the plant at instant k_0 (this input is the first entry of the optimal solution $U^{\star}_{k_0}$ at time k_0), it follows that:

$$\begin{aligned} \Delta J(k_0) &= J(k_0+1) - J(k_0) \\ &= -x(k_0+1)^T Q x(k_0+1) \\ &- (u(k_0)^{\star})^T R u^{\star}(k_0) + V(x(k_0+1+N_p) \\ &+ x(k_0+1+N_p)^T Q x(k_0+1+N_p) \\ &+ x(k_0+1+N_p) K^{\mathrm{n}}(\rho)^T R K^{\mathrm{n}}(\rho) x(k_0+1+N_p) \\ &- V(x(k_0+N_p)) . \end{aligned} \tag{35}$$

From (C3), we can pursue with the negativeness of the following term:

$$\begin{aligned} 0 &\geq V(x(k_0+N_p+1)) - V(x(k_0+N_p)) \\ &+ x(k_0+N_p)^T \left(Q + (K^{\mathrm{n}}(\rho))^T R K^{\mathrm{n}}(\rho) \right) x(k_0+N_p) . \end{aligned} \tag{36}$$

[4]We use compact notation for brevity. Herein, we use $\rho := \rho(k_0+1)$.

Thus, substituting this inequality into Eq. (35) yields:

$$\begin{aligned} \Delta J(k_0) &\leq -x(k_0+1)^T Q x(k_0+1) \\ &- (u(k_0)^\star)^T R u^\star(k_0)\,. \end{aligned} \tag{37}$$

Since Q and R are positive defined matrices by construction (they are the MPC tuning weights) and $u^\star(k_0)$ and $x(k_0+1)$ are known[5], it follows that:

$$\Delta J(k_0) \leq 0\,, \tag{38}$$

which means that the MPC cost function decays along k.

From the optimality of the solution $U^\star_{k+1}$, it follows that the cost constructed with this sequence ($J^\star(k_0+1)$) holds as a lower-bound with respect to $J(k_0+1)$, this is $J^\star(k_0+1) \leq J(k_0+1)$. Then, using this argument on $\Delta J(k_0)$, we arrive at:

$$\begin{aligned} J(k_0+1) - J(k_0) &\leq 0\,, \\ J^\star(k_0+1) &\leq J(k_0+1) \leq J(k_0)\,, \end{aligned} \tag{39}$$

which proves that $J^\star$ is a Lyapunov function and x will converge to the origin (due to (C1)), as long as the initial condition provides, in fact, a feasible starting point. We note that (C2) is necessary to map a feasible $x(k_0+1|k_0)$. Conditions (C4) and (C5) are necessary to ensure admissible trajectories regarding x and u.

[5]Note that $x(k_0+1) = A(\rho(k_0))x(k_0) + B(\rho(k_0))u^\star(k_0)$.