Received: 25 August 2022 | Revised: 15 November 2022 | Accepted: 12 January 2023

DOI: 10.1049/cth2.12430

IET Control Theory & Applications

The Institution of Engineering and Technology WILEY

ORIGINAL RESEARCH

Efficient implementation of Gaussian process–based predictive control by quadratic programming

Péter Polcz[1,2] | Tamás Péni[1] | Roland Tóth[1,3]

[1]Systems and Control Laboratory, Institute for Computer Science and Control, Budapest, Hungary

[2]Faculty of Information Technology and Bionics, Pázmány Péter Catholic University, Budapest, Hungary

[3]Control Systems Group, Department of Electrical Engineering, Eindhoven University of Technology, Eindhoven, The Netherlands

Correspondence
Tamás Péni, Systems and Control Laboratory, Institute for Computer Science and Control, Budapest, Hungary.
Email: peni@sztaki.hu

Funding information
European Union, Grant/Award Number: RRF-2.3.1-21-2022-00002; Eötvös Loránd Research Network, Grant/Award Number: SA-77/2021

Abstract

The paper addresses the problem of accelerating predictive control of non-linear system models augmented with Gaussian processes (GP-MPC). Due to the non-linear and stochastic prediction model, predictive control of GP-based models requires to solve a stochastic optimization problem. Different model simplification methods have to be applied to reformulate this problem to a deterministic, non-linear optimization task that can be handled by a numerical solver. As these problems are still complex, especially with exact moment calculations, real-time implementation of GP-MPC is extremely challenging. The existing solutions accelerate the computations at the solver level by linearizing the non-linear optimization problem and applying sequential convexification. In contrast, this paper proposes a novel GP-MPC solution approach that without linearization formulates a series of surrogate quadratic programs (QP-s) to iteratively obtain the solution of the original non-linear optimization problem. The first step is embedding the non-linear mean-variance dynamics of the GP-MPC prediction model in a linear parameter-varying (LPV) structure and rewriting the constraints in parameter-varying form. By fixing the scheduling trajectory at a known variation (based on previously computed or initial state-input trajectories), optimization of the input sequence for the remaining varying linear model reduces to a linearly constrained quadratic program. After solving the QP, the non-linear prediction model is simulated for the new control input sequence and new scheduling trajectories are updated. The procedure is iterated until the convergence of the scheduling, that is, the solution of the QP converges to the solution of the original non-linear optimization problem. By designing a reference tracking controller for a 4DOF robot arm, we illustrate that the convergence is remarkably fast and the approach is computationally advantageous compared to current solutions. The proposed method enables the application of GP-MPC algorithms even with exact moment matching on fast dynamical systems and requires only a QP solver.

1 | INTRODUCTION

In the past two decades, model learning approaches gained an increased interest in a variety of control applications [1], as they provide efficient adaptation to variation or unknown aspects of the dynamical behaviour through measurements.

Among the various approaches of machine learning, probabilistic *Gaussian process* (GP) models have the advantage of being able to describe efficiently difficult function relationships and adapt quickly to continuously changing relations. Moreover, a GP can indicate the quality of the prediction through its computed variance, which can be used to characterize the uncertainty of the model. A GP is fully described by a mean function and a covariance function [2], the latter also qualifies as a kernel function in the associated *reproducing kernel Hilbert space* (RKHS), allowing to characterize optimally of the GP as a function estimator [3].

In learning-based control design, GPs often appear as an additive augmentation term beside the nominal dynamical model of the system and their role is to learn/capture the unknown part of the dynamics from measurement data [4, 5]. Although this structure is favourable for adaptive modelling and the uncertainty of the GP-based model can be used to characterize stability guarantees for the true underlying system despite modelling errors [6–8], the control design is challenging, because the GP adds a non-linear stochastic component to the nominal system. This is especially true for *GP-based model predictive control* (GP-MPC), where a future prediction of the state has to be constructed by recursive application of the augmented model. The distribution of the predicted future state becomes more and more complex as the time index goes forward, therefore the control design results in a non-linear stochastic optimization problem. Series of approximation steps have to be carried out to transform the original stochastic optimization problem to a deterministic one.

The majority of the GP-MPC implementations rely on the assumption that at every prediction time step, the state distribution can be well approximated by a normal distribution [9–13]. A few kernel functions even make it possible to compute the first two moments of the predicted output distribution analytically if the GP input is normally distributed [11–14]. This technique is often referred to as *exact moment* (EM) matching. Alternatively, the state distribution can be approximated by using the first-order part of the Taylor expansion of the predictive mean function and the mean equivalent approximation of the variance function [15]. Although these techniques result in *deterministic* mean-variance prediction models, which enable implementation of GP-MPC in practice, they still require the solution of a difficult non-linear optimization problem.

Finding an optimal solution to the resulting *non-linear MPC* (NMPC) is still cumbersome, and it is often approximated iteratively by solving a sequence of convex quadratic problems [16–18]. These approaches are based on a linearization of the optimization problem around a reference solution, and require the differentiation of the dynamic prediction model. Similar methods are available for non-linear systems with stochastic error processes [19, 20] or GP-based stochastic autoregressive models based on predictive control [21, 22]. However, the method of gradients may become computationally too intensive, when the moments of the GP-based state-prediction are computed with EM matching. Hence, often only the Taylor approximation is used at the expanse of increased approximation error [12, 22].

Inspired by the *linear-parameter varying* (LPV) scheme for deterministic NMPC introduced in [23, 24], our main contribution includes the following:

C1 Novel iterative method for rapid solution of GP-MPC problems. Our developed approach is applicable for discrete-time non-linear state-space models additively augmented by GPs. To establish it, the following sub-contributions of this paper are important.

C2 LPV embedding of the GP-MPC problem both under Taylor and exact moment-based approximations of the stage propagation.

C3 Recasting the embedded GP-MPC problem as a computationally cheap LPV problem (quadratic program with linear constraints).

C4 Developing an iterative scheme where previous LPV solutions are used to converge to the optimal solution of the GP-MPC problem.

C5 Thorough testing of the developed scheme in simulation studies.

In terms of C2, the mean-variance prediction dynamics associated with the GP-MPC problem are embedded in an LPV form that makes available linear prediction of the state-dynamics along a fixed, time-varying scheduling sequence and conversion of the state and input constraints to linear scheduling-dependent form. This is directly needed to realize C3. For C4, the computed sequence of optimal inputs from the LPV-MPC problem is used to simulate the non-linear mean-variance prediction model. Then the scheduling trajectory is updated by the computed inputs and by the simulated state distribution to iteratively re-solve the LPV-MPC problem until convergence of the obtained solution occurs. Note that the approach is proposed with both EM matching and Taylor-based mean-variance prediction models and provides fast convergence to an optimum of the GP-MPC problem with lower computational cost than the available state-of-the-art solutions. The significance of our contributions C1-C4 using this iterated LPV-MPC form is in providing a simple yet efficient technique for solving large-scale GP-MPC problems even under EM-based prediction models with simple QPs and without compromising accuracy.

The paper is organized as follows: the problem setting and important notions, such as GP-based model augmentation, are discussed first in Section 2. Then, we present an overview of the existing GP-MPC formulation and approximation results in Section 3. As our main contribution (C1), the proposed LPV acceleration approach for GP-MPC is presented in Section 4 detailing the required ingredients in terms of C2–C4. We illustrate the effectiveness of our approach in simulation studies in Section 5 corresponding to C5. Finally, the main conclusions on the presented approach are drawn in Section 6.

Notation 1. Let $\mathbb{Z}$, $\mathbb{N}$ and $\mathbb{R}$ denote the set of integers, non-negative integers and real numbers, respectively, while $\mathbb{N}_+ = \mathbb{N} \setminus 0$ and $\mathbb{R}_+$ is the set of non-negative reals. We denote by $\mathbb{S}_+^n$ the set of symmetric positive definite matrices in $\mathbb{R}^{n\times n}$. Let $\operatorname{vec}(x_1, \dots, x_n) = [\, x_1^\top \ \cdots \ x_n^\top \,]^\top$ denote the column-wise composition of vectors $x_\tau \in \mathbb{R}^n$, while $\operatorname{diag}(x_1, \dots, x_n)$ corresponds to block diagonal matrix composition. Consider a time series $k \mapsto x(k) \in \mathbb{R}^n$, where $k \in \mathbb{N}$ is the discrete time. Let $x(i|k)$ denote the value of $x(k+i)$ predicted at time k. When it is convenient, we suppress the time argument of $x(k)$ representing the whole series as x. Notation x^+ refers to the forward shifted sequence of x, that is, $x^+(k) = x(k+1)$ or $x^+(i|k) = x(i+1|k)$. For two random variables x and y, taking values in $\mathbb{R}^n$, the expected value and variance matrix of x are denoted by $\mathbb{E}(x)$ and $\operatorname{Var}(x)$, while the covariance matrix of x and y is $\operatorname{Cov}(x, y)$.

When x is normally distributed with expected value μ and variance Σ, we write $x \sim \mathcal{N}(\mu_{\mathrm{x}}, \Sigma_{\mathrm{x}})$. We say that $\mathcal{X}$ is a credibility set for x with probability level p_{x} if $\Pr(x \in \mathcal{X}) \geq p_{\mathrm{x}}$. We denote the Kronecker delta function as $\delta(x, \tilde{x})$ with $\delta(x, x) = 1$ and $\delta(x, \tilde{x}) = 0$ otherwise. We also denote a set of indices as $\mathbb{I}_{\tau_1}^{\tau_2} = \{s \in \mathbb{Z} \mid \tau_1 \leq s \leq \tau_2\}$. Note that in our notation, non-italic letters in the subscript, such as in μ_{x}, are labels that express that the coefficient or variable relates to an other variable, for example, μ_{x} is the expectation of vector x, while italic letters express that the variable depends on an other variable, like κ_i expresses the ith element of κ. The textual superscripts in notation like $\mu_{\mathrm{z}}^{\mathrm{EM}}$ are abbreviations, which denote different definitions or versions of the same variable.

2 | MODEL AUGMENTATION

2.1 | System model

As the plant to be controlled, we consider a discrete-time nonlinear system in the following form:

$$x^+ = f(x, u) + g_{\mathrm{d}}(x, u) + v, \tag{1a}$$

$$y = Cx, \tag{1b}$$

where $x(k) \in \mathbb{R}^{n_{\mathrm{x}}}$, $u(k) \in \mathbb{R}^{n_{\mathrm{u}}}$, $y(k) \in \mathbb{R}^{n_{\mathrm{y}}}$, and $v(k) \in \mathbb{R}^{n_{\mathrm{x}}}$ are the state, control input, measured output and external disturbance signals, respectively, while $f : \mathbb{R}^{n_{\mathrm{x}} \times n_{\mathrm{u}}} \to \mathbb{R}^{n_{\mathrm{x}}}$ and $g_{\mathrm{d}} : \mathbb{R}^{n_{\mathrm{x}} \times n_{\mathrm{u}}} \to \mathbb{R}^{n_{\mathrm{x}}}$ are bounded deterministic vector functions. The function f constitutes the physically well-interpretable and a priori known dynamics of the system, that is, its *nominal model*, whereas, g_{d} represents the unknown, that is, *unmodelled* dynamics of the system. To simplify the notation, we introduce $w = \mathrm{vec}(x, u)$ to denote the arguments of these functions. In terms of the measured output y, we assume that (1b) is linear with a constant $C \in \mathbb{R}^{n_{\mathrm{y}} \times n_{\mathrm{x}}}$. The external disturbance v is assumed to be an *independent and identically distributed* (i.i.d.) white noise process with variance $\Sigma_{\mathrm{v}} = \mathrm{diag}(\sigma_{\mathrm{v},1}^2, \ldots, \sigma_{\mathrm{v},n_{\mathrm{x}}}^2)$. We moreover assume that the full state vector is available for measurement.

Our objective is twofold, (i) next to efficiently estimate the unknown part of the dynamics g_{d} from data together with reliable uncertainty bounds, (ii) we aim to design a reference tracking model predictive controller for the resulting estimated model of (1) which can provide computationally efficient control of (1) with stability and performance guarantees. Step (i) is required to provide a prediction model for (ii) that is able to generate a prediction $x(1|k), \ldots, x(N_{\mathrm{p}}|k)$ of the future state trajectory as the response of the system for the initial state $x(0|k) = x(k)$ and the future control sequence $u(k), u(k+1), \ldots, u(k+N_{\mathrm{p}}-1)$. Here $N_{\mathrm{p}} > 0$ is the prediction horizon, and $k \geq 0$ is any time instant.

2.2 | Augmentation by a Gaussian process

The major challenge in establishing the predictor for (ii) is that the system model (1) is only partially known, therefore, the unknown dynamic component $z = x^+ - f(w) = g_{\mathrm{d}}(w) + v$ has to be estimated by using a collection of measurement data $\mathcal{D}_N = \{(w(k), z(k))\}_{k=1}^N$ generated by (1). To identify an unknown mapping $w \in \mathbb{R}^{n_{\mathrm{w}}} \to z \in \mathbb{R}^{n_{\mathrm{z}}}$, where $n_z = n_{\mathrm{x}}$, efficiently with a characterization of the remaining uncertainty of the estimate, we use GP regression [2, 13]. To distinguish w from the control input u, and z from the system output y, we call w and z as the *GP-input* and *GP-output*, respectively.

In terms of definition, a vectorial *Gaussian Process* $\mathcal{GP} : \mathbb{R}^{n_{\mathrm{w}}} \to \mathbb{R}^{n_{\mathrm{z}}}$ assigns to every point $w \in \mathbb{R}^{n_{\mathrm{w}}}$ a random variable $\mathcal{GP}(w)$ taking values in $\mathbb{R}^{n_{\mathrm{z}}}$ such that, for any finite set $\{w(\tau)\}_{\tau=1}^N \subset \mathbb{R}^{n_{\mathrm{w}}}$, the joint probability distribution of $\mathcal{GP}(w(1)), \ldots, \mathcal{GP}(w(N))$ is multidimensional Gaussian. GPs are fully determined by their mean m and covariance functions κ, hence if $g \sim \mathcal{GP}(m, \kappa)$,

$$m(w) = \mathbb{E}\{g(w)\}$$

$$\kappa(w, \tilde{w}) = \mathbb{E}\{(g(w) - m(w))(g(\tilde{w}) - m(\tilde{w}))^\top\},$$

Both m and κ, where the latter is also called a *kernel* function, are often parameterized in terms of *hyperparameters* $\theta \in \mathbb{R}^{n_\theta}$. In fact, taking $g \sim \mathcal{GP}(m, \kappa)$ as the prior distribution in the estimation process defines the prior knowledge about g_{d} in terms of the mean function m. On the other hand, the choice of κ determines the function space in which an estimate of the function g_{d} is searched for together with the prior model of the noise process. Parameterization of m and κ in terms of *hyperparameters* θ allows to adjust the prior, that is, these choices to the estimation problem of g_{d} using $\mathcal{D}_N$.

Furthermore, GP regression is often implemented for each element of the GP-output variable separately, i.e, in terms of scalar-valued $g_i \sim \mathcal{GP}(m_i, \kappa_i)$ with $i \in \mathbb{I}_1^{n_{\mathrm{z}}}$ constituting to $g = \mathrm{vec}(g_1, \ldots, g_{n_z})$. While this approach greatly simplifies the estimation problem, it inherently results in the choice of $\kappa = \mathrm{diag}(\kappa_1, \ldots, \kappa_{n_z})$.

Various types of kernels are used in applications, for example, the Matérn class, the exponential family, the dot product family, periodic kernels, and so on, or their combinations [2, Sec. 4.2]. Due to its simplicity and advantageous properties regarding the *moment matching* concept we will exploit later on, we consider the scalar *squared exponential* (SE) kernel for the modelling of g_{d}:

$$\kappa_{\sigma_{\mathrm{g}},\Lambda}^{\mathrm{SE}}(w, \tilde{w}) = \sigma_{\mathrm{g}}^2 \exp\left(-\frac{1}{2}(w - \tilde{w})^\top \Lambda^{-1}(w - \tilde{w})\right), \tag{2}$$

where $\sigma_{\mathrm{g}} \in \mathbb{R}_+$ and $\Lambda \in \mathbb{R}^{n_{\mathrm{w}} \times n_{\mathrm{w}}}$ is positive definite. By taking into account the assumed i.i.d. properties of v and that its variance is $\Sigma_{\mathrm{v}} = \mathrm{diag}(\sigma_{\mathrm{v},1}^2, \ldots, \sigma_{\mathrm{v},n_{\mathrm{x}}}^2)$, the kernel κ_i for each $g_i \sim \mathcal{GP}(m_i, \kappa_i)$ is formulated as

$$\kappa_i(w, \tilde{w}) = \kappa_{\sigma_{\mathrm{g},i},\Lambda_i}^{\mathrm{SE}}(w, \tilde{w}) + \delta(w, \tilde{w})\sigma_{\mathrm{v},i}^2. \tag{3}$$

With the choice of (3) and with a zero prior mean ($m_i \equiv 0$), commonly used in the literature [2, Sec. 2.7], $\theta = \mathrm{vec}(\{\sigma_{\mathrm{g},i}, \Lambda_i, \sigma_{\mathrm{v},i}\}_{i=1}^{n_{\mathrm{z}}})$ are the tunable *hyperparameters* of the GP model and often optimized by the maximization of the marginal likelihood of z under θ and the observed $\mathcal{D}_N$ [2, Sec. 5.4.1].

Based on the given $\mathcal{D}_N$ and the prior $g_i \sim \mathcal{GP}(0, \kappa_i)$,

$$p(Z_i | W, \theta) = \mathcal{N}(0, K_{w,w,i}), \tag{4}$$

where $[K_{w,w,i}]_{\tau,\tilde{\tau}} = \kappa_i(w(\tau), w(\tilde{\tau}))$ with $\tau, \tilde{\tau} \in \mathbb{I}_1^N$, describes the probability density function of the outputs $Z_i = [\, z_i(1) \ \cdots \ z_i(N) \,]^\top$ seen as random variables conditioned on the observed inputs $W = [\, w(1) \ \cdots \ w(N) \,]^\top$ and hyper-parameter values θ.

To predict the value of z_i at a test point w_*, the following joint distribution

$$\begin{bmatrix} Z_i \\ z_{*,i} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K_{w,w,i} & K_{w,i}(w_*) \\ K_{w,i}^\top(w_*) & \kappa_i(w_*, w_*) \end{bmatrix} \right)$$

with $[K_{w,i}(w_*)]_\tau = \kappa_i(w(\tau), w_*)$ holds based on the previous considerations. Hence, the predictive distribution for $z_{*,i}$, based on the observed samples $\{z_i(\tau)\}_{\tau=1}^N$ in $\mathcal{D}_N$, is the posteriori $p(z_{*,i} | \mathcal{D}_N, w_*) = \mathcal{N}(\mu_{z,i}^{\mathrm{GP}}(x_*), \sigma_{z,i}^{\mathrm{GP}}(w_*, w_*))$ characterized by

$$\mu_{z,i}^{\mathrm{GP}}(w_*) = \overbrace{K_{w,i}^\top(w_*) K_{w,w,i}^{-1}}^{\beta_i} Z_i \tag{5a}$$

$$\sigma_{z,i}^{\mathrm{GP}}(w_*, \tilde{w}_*) = \kappa_i(w_*, \tilde{w}_*) - K_{w,i}^\top(w_*) K_{w,w,i}^{-1} K_{w,i}(\tilde{w}_*). \tag{5b}$$

Computation of (5) requires only elementary matrix operations, therefore it is computationally efficient. In terms of the complete predictive distribution of z, the mean $\mu_z^{\mathrm{GP}}(w_*)$ with $\mu_z^{\mathrm{GP}} = \mathrm{vec}(\mu_{z,1}^{\mathrm{GP}}, \dots, \mu_{z,n_z}^{\mathrm{GP}})$ gives an approximation of $g_{\mathrm{d}}(w_*)$ while the variance $\Sigma_z^{\mathrm{GP}}(w_*, w_*)$ with $\Sigma_z^{\mathrm{GP}} = \mathrm{diag}(\sigma_{z,1}^{\mathrm{GP}}, \dots, \sigma_{z,n_z}^{\mathrm{GP}})$ gives a measure of the uncertainty of this approximation together with the added uncertainty of v.

By using the resulting predictive distribution to define $\mathcal{GP}_z = \mathcal{GP}(\mu_z^{\mathrm{GP}}, \Sigma_z^{\mathrm{GP}})$ that approximates the unknown term $g(w) + v$ in (1), the prediction model can be written as follows:

$$\hat{x}^+ = f(w) + \underbrace{\mathcal{GP}_z(w)}_{\hat{z}}. \tag{6}$$

Recursive application of this model requires the evaluation of f and the $\mathcal{GP}_z$ at uncertain w as the state transition predicted in a previous step results in a Gaussian distribution of possible values. If w is a random variable and f is non-linear, then the distribution of $f(w)$ and the posterior distribution of the GP are both non-Gaussian, which makes the computation often intractable. To overcome this problem, various approximations have been introduced in the literature, which are overviewed in the next subsection.

2.3 | Model approximation

To simplify the model (6) and express the predicted states in an efficiently computable closed form, which is a function of the control input, the general practice is to approximate every predicted future state $x(i|k)$, $i \in \mathbb{I}_1^{N_{\mathrm{p}}}$ by a Gaussian random variable, where $N_{\mathrm{p}} \in \mathbb{N}_+$ is the *prediction horizon*. If the control input is also deterministic or Gaussian (e.g., linear function of the state), then w in (6) is Gaussian as well. Hence, based on the Gaussian assumption of w, we need to construct a Gaussian approximation for x^+ in (6) to provide an efficient recursively computable approximation of the distribution of $x(i|k)$. This section gives an overview of the most commonly used approaches for this purpose.

First, note that the nominal and the disturbance models in (6) are different in their nature. The nominal model $f(w)$ is often a general non-linear term without probabilistic components, whereas the Gaussian process $\mathcal{GP}_z(w)$ represents a stochastic term with a fixed non-linear structure. Therefore, the two terms are often handled separately in the approximation schemes.

The distribution of $f(w)$ can be approximated by computing the Taylor series of f around μ_{w} and keeping only the affine terms:

$$f(w) \approx f(\mu_{\mathrm{w}}) + \frac{\partial f}{\partial w}(\mu_{\mathrm{w}})\,(w - \mu_{\mathrm{w}}) \tag{7}$$

Clearly, if w is Gaussian, that is, $w \sim \mathcal{N}(\mu_{\mathrm{w}}, \Sigma_{\mathrm{w}})$, the right-hand side of (7) defines a Gaussian random variable.

Next, to characterize the stochastic properties of x^+, the distribution of $f(w) + \hat{z}$ with $\hat{z} = \mathcal{GP}_z(w)$ has to be determined. This can be done by computing the joint distribution of the summed random variables. Since $f(w)$ is approximated by the linear mapping (7), only the joint distribution of w and $\hat{z}$ is needed to be determined to answer this question. To get a normal distribution for x^+, the joint distribution is approximated by a Gaussian one:

$$\begin{bmatrix} w \\ \hat{z} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_{\mathrm{w}} \\ \mu_z(\mu_{\mathrm{w}}, \Sigma_{\mathrm{w}}) \end{bmatrix}, \begin{bmatrix} \Sigma_{\mathrm{w}} & \Sigma_{\mathrm{wz}}(\mu_{\mathrm{w}}, \Sigma_{\mathrm{w}}) \\ \star & \Sigma_z(\mu_{\mathrm{w}}, \Sigma_{\mathrm{w}}) \end{bmatrix} \right). \tag{8}$$

Despite the conceptual simplicity of (8), the moments of (8) are still challenging to determine. Various approaches have been introduced to compute or at least approximate the values of μ_z, Σ_z, and Σ_{wz}. Next, we present two of these techniques.

Taylor approximation: One may assume that the mean and the variance of the GP are separable, namely, $\mathcal{GP}_z(w) \approx \mu_z^{\mathrm{GP}}(w) + \mathcal{N}(0, \Sigma_z^{\mathrm{GP}}(w))$. Then, the first-order Taylor approximation of the mean function μ_z^{GP}, and the mean equivalent approximation of the variance function Σ_z^{GP} gives

$$\mu_z(\mu_{\mathrm{w}}, \Sigma_{\mathrm{w}}) \approx \mu_z^{\mathrm{GP}}(\mu_{\mathrm{w}}) \tag{9a}$$

$$\begin{bmatrix} \Sigma_{\mathrm{wz}}(\mu_{\mathrm{w}}, \Sigma_{\mathrm{w}}) \\ \Sigma_z(\mu_{\mathrm{w}}, \Sigma_{\mathrm{w}}) \end{bmatrix} \approx \begin{bmatrix} \Sigma_{\mathrm{w}} \left(\dfrac{\partial \mu_z^{\mathrm{GP}}}{\partial w}(\mu_{\mathrm{w}}) \right)^\top \\ \Sigma_z^{\mathrm{GP}}(\mu_{\mathrm{w}}) + \left(\dfrac{\partial \mu_z^{\mathrm{GP}}}{\partial w}(\mu_{\mathrm{w}}) \right) \Sigma_{\mathrm{w}} \left(\dfrac{\partial \mu_z^{\mathrm{GP}}}{\partial w}(\mu_{\mathrm{w}}) \right)^\top \end{bmatrix}. \tag{9b}$$

$$\mu^{\mathrm{EM}}_{\mathrm{z},i}(\mu_{\mathrm{w}},\Sigma_{\mathrm{w}}) = \beta_i^{\top} q_i(\mu_{\mathrm{w}},\Sigma_{\mathrm{w}}), \tag{10a}$$

$$\Sigma^{\mathrm{EM}}_{\mathrm{z},i,i}(\mu_{\mathrm{w}},\Sigma_{\mathrm{w}}) = \beta_i^{\top} \mathcal{Q}_{i,i}(\mu_{\mathrm{w}},\Sigma_{\mathrm{w}})\,\beta_i - \left(\mu^{\mathrm{EM}}_{\mathrm{z},i}(\mu_{\mathrm{w}},\Sigma_{\mathrm{w}})\right)^2 + \sigma^2_{\mathrm{g},i} - \mathrm{Tr}(K^{-1}_{w,w,i}\,\mathcal{Q}_{i,i}(\mu_{\mathrm{w}},\Sigma_{\mathrm{w}})), \tag{10b}$$

$$\Sigma^{\mathrm{EM}}_{\mathrm{z},i,j}(\mu_{\mathrm{w}},\Sigma_{\mathrm{w}}) = \beta_i^{\top} \mathcal{Q}_{i,j}(\mu_{\mathrm{w}},\Sigma_{\mathrm{w}})\,\beta_j - \mu^{\mathrm{EM}}_{\mathrm{z},i}(\mu_{\mathrm{w}},\Sigma_{\mathrm{w}})\,\mu^{\mathrm{EM}}_{\mathrm{z},j}(\mu_{\mathrm{w}},\Sigma_{\mathrm{w}}), \quad \text{if } i \neq j, \tag{10c}$$

$$\Sigma^{\mathrm{EM}}_{\mathrm{wz},:,i}(\mu_{\mathrm{w}},\Sigma_{\mathrm{w}}) = \textstyle\sum_{\tau=1}^{N} \beta_{i,\tau}\, q_{i,\tau}(\mu_{\mathrm{w}},\Sigma_{\mathrm{w}})\,\Sigma_{\mathrm{w}}(\Sigma_{\mathrm{w}}+\Lambda_i)^{-1}(w(\tau)-\mu_{\mathrm{w}}), \tag{10d}$$

$$\text{where } q_{i,\tau}(\mu_{\mathrm{w}},\Sigma_{\mathrm{w}}) = \sigma^2_{\mathrm{g},i}\det\left(\Sigma_{\mathrm{w}}\Lambda_i^{-1}+I_{n_{\mathrm{w}}}\right)^{-\frac{1}{2}}\exp\left(-\tfrac{1}{2}(w(\tau)-\mu_{\mathrm{w}})^{\top}(\Sigma_{\mathrm{w}}+\Lambda_i)^{-1}(w(\tau)-\mu_{\mathrm{w}})\right), \tag{10e}$$

$$[\mathcal{Q}_{i,j}(\mu_{\mathrm{w}},\Sigma_{\mathrm{w}})]_{\tau,\tilde{\tau}} = \det(R_{i,j}(\Sigma_{\mathrm{w}}))^{-\frac{1}{2}}\,\sigma^2_{\mathrm{g},i}\,\sigma^2_{\mathrm{g},j}\,[\breve{\mathcal{Q}}_{i,j}(\mu_{\mathrm{w}},\Sigma_{\mathrm{w}})]_{\tau,\tilde{\tau}} \tag{10f}$$

$$[\mathcal{Q}_{i,j}(\mu_{\mathrm{w}},\Sigma_{\mathrm{w}})]_{\tau,\tilde{\tau}} = \exp\left(\frac{1}{2}\xi^{\top}_{\tau,\tilde{\tau}}(\mu_{\mathrm{w}})R^{-1}_{i,j}(\Sigma_{\mathrm{w}})\,\Sigma_{\mathrm{w}}\,\xi_{\tau,\tilde{\tau}}(\mu_{\mathrm{w}}) - \frac{1}{2}\zeta^{\top}_{\tau}(\mu_{\mathrm{w}})\Lambda_i^{-1}\zeta_{\tau}(\mu_{\mathrm{w}}) - \frac{1}{2}\zeta^{\top}_{\tilde{\tau}}(\mu_{\mathrm{w}})\Lambda_j^{-1}\zeta_{\tilde{\tau}}(\mu_{\mathrm{w}})\right), \tag{10g}$$

$$R_{i,j}(\Sigma_{\mathrm{w}}) = \Sigma_{\mathrm{w}}(\Lambda_i^{-1}+\Lambda_j^{-1}) + I_{n_{\mathrm{w}}}, \quad \zeta_{\tau}(\mu_{\mathrm{w}}) = w(\tau)-\mu_{\mathrm{w}}, \xi_{\tau,\tilde{\tau}}(\mu_{\mathrm{w}}) = \Lambda_i^{-1}\zeta_{\tau}(\mu_{\mathrm{w}}) + \Lambda_j^{-1}\zeta_{\tilde{\tau}}(\mu_{\mathrm{w}}), \tag{10h}$$

for each $i, j \in \mathbb{I}_1^{n_{\mathrm{z}}}$ and $\tau, \tilde{\tau} \in \mathbb{I}_1^{N}$.

Exact moment matching. According to [11, 13], the squared exponential kernel (3) makes it possible to determine the first two moments (10) of the joint distribution (8) analytically: $\mu_{\mathrm{z}} = \mu^{\mathrm{EM}}_{\mathrm{z}}$, $\Sigma_{\mathrm{z}} = \Sigma^{\mathrm{EM}}_{\mathrm{z}}$, $\Sigma_{\mathrm{wz}} = \Sigma^{\mathrm{EM}}_{\mathrm{wz}}$. Compared to the Taylor approximation, the *exact moment* (EM) matching gives a more precise Gaussian approximation of the actual distribution of the Gaussian process $\mathcal{GP}_{\mathrm{z}}(w)$ at a non-deterministic query point w. However, analytic moment calculations require heavy matrix operations and involve complicated non-linear expressions (10) of the mean and the variance of w.

2.4 | Approximative non-linear prediction model

The introduced approximation approaches provide a rolling computation of the predicted states:

$$x(i|k) \approx \mathcal{N}(\mu_{\mathrm{x}}(i|k), \Sigma_{\mathrm{x}}(i|k)), \tag{11}$$

where the mean and the variance are computed as

$$\mu_{\mathrm{x}}(i+1|k) = A\left(\mu_{\mathrm{w}}(i|k)\right)\mu_{\mathrm{w}}(i|k) + c\left(\mu_{\mathrm{w}}(i|k), \Sigma_{\mathrm{w}}(i|k)\right), \tag{12a}$$

$$\Sigma_{\mathrm{x}}(i+1|k) = [A\left(\mu_{\mathrm{w}}(i|k)\right)\ I] \cdot \begin{bmatrix} \Sigma_{\mathrm{w}}(i|k) & \Sigma_{\mathrm{wz}}(\mu_{\mathrm{w}}(i|k), \Sigma_{\mathrm{w}}(i|k)) \\ \star & \Sigma_{\mathrm{z}}(\mu_{\mathrm{w}}(i|k), \Sigma_{\mathrm{w}}(i|k)) \end{bmatrix} [A\left(\mu_{\mathrm{w}}(i|k)\right)\ I]^{\top} \tag{12b}$$

where coefficient functions A and c depend on the actual approximation technique (Taylor or EM) applied for (8). In particular, the *Taylor approximation* of the predictive mean function $\mu^{\mathrm{GP}}_{\mathrm{z}}$ gives

$$A(\mu_{\mathrm{w}}) = \frac{\partial f}{\partial w}(\mu_{\mathrm{w}}) + \frac{\partial \mu^{\mathrm{GP}}_{\mathrm{z}}}{\partial w}(\mu_{\mathrm{w}}) \tag{13a}$$

$$c(\mu_{\mathrm{w}}, \Sigma_{\mathrm{w}}) = f(\mu_{\mathrm{w}}) + \mu^{\mathrm{GP}}_{\mathrm{z}}(\mu_{\mathrm{w}}) - \frac{\partial f}{\partial w}(\mu_{\mathrm{w}})\mu_{\mathrm{w}} - \frac{\partial \mu^{\mathrm{GP}}_{\mathrm{z}}}{\partial w}(\mu_{\mathrm{w}})\mu_{\mathrm{w}}, \tag{13b}$$

the EM approach results in

$$A(\mu_{\mathrm{w}}) = \frac{\partial f}{\partial w}(\mu_{\mathrm{w}}) \tag{14a}$$

$$c(\mu_{\mathrm{w}}, \Sigma_{\mathrm{w}}) = f(\mu_{\mathrm{w}}) - \frac{\partial f}{\partial w}(\mu_{\mathrm{w}})\,\mu_{\mathrm{w}} + \mu^{\mathrm{EM}}_{\mathrm{z}}(\mu_{\mathrm{w}}, \Sigma_{\mathrm{w}}). \tag{14b}$$

3 | THE PREDICTIVE CONTROL PROBLEM

In this section, we formulate a model predictive control design problem for system (6). By applying the introduced approximation schemes and (12), this problem can be rewritten to a deterministic non-linear optimization task. This section is intended to introduce this non-linear optimization problem, which we reformulate in Section 4 to be solved efficiently in terms of QP iterations both under the Taylor and EM approximation schemes.

We consider a reference tracking control problem, where a time-varying reference signal $r(k) \in \mathbb{R}^{n_\mathrm{y}}$ is required to be tracked by the system output y. Following the MPC concept, the controller seeks at each time instant k, a sequence of control inputs $\{u(i|k)\}_{i=0}^{N_\mathrm{p}-1}$ by 'minimizing' the cost

$$J_k = \sum_{i=1}^{N_\mathrm{p}} \|r(k+i) - Cx(i|k)\|_Q^2 + \sum_{i=0}^{N_\mathrm{p}-1} \|u(i|k)\|_R^2. \tag{15}$$

Here $\|\cdot\|_Q$ and $\|\cdot\|_R$ denote the weighted 2-norm with respect to symmetric positive definite matrices $Q \in \mathbb{R}^{n_\mathrm{y}\times n_\mathrm{y}}$ and $R \in \mathbb{R}^{n_\mathrm{u}\times n_\mathrm{u}}$ and $N_\mathrm{p} \in \mathbb{N}_+$ is the prediction horizon. As the state and input can be both random variables, the cost function (15) is inherently probabilistic. Therefore, an often-used strategy is to minimize the 'most likely,' that is, *expected value* of the cost (15): $\mathbb{E}(J_k)$.

In physical systems, the states and inputs are often restricted to an operating region, which can be expressed in terms of sets $\mathcal{X} \subseteq \mathbb{R}^{n_\mathrm{x}}$, $\mathcal{U} \subseteq \mathbb{R}^{n_\mathrm{u}}$. Hence, next to the minimization of the expectation of (15), we can formulate the satisfaction of state and input constraints in a probabilistic sense

$$\Pr(x(i+1|k) \in \mathcal{X}) \geq p_\mathrm{x},\ \Pr(u(i|k) \in \mathcal{U}) \geq p_\mathrm{u}, \tag{16}$$

where p_x and p_u are given probability levels of the credibility sets.

We assume that the full state vector is available for measurement without any noise at every control cycle

$$x(0|k) \sim \mathcal{N}\left(\mu_\mathrm{x}(k), \Sigma_{\mathrm{x},0}\right), \tag{17}$$

where $\mu_\mathrm{x}(k) = x(k)$ and the covariance $\Sigma_{\mathrm{x},0} = 0$.

The stochastic non-linear model (6) provides distributed predictions for the future states $x(i|k)$, $i \in \mathbb{I}_1^{N_\mathrm{p}}$. Due to the recursive construction, the uncertainty of the predicted states accumulates as the time index goes forward in time. This may result in large state distributions towards the end of the horizon, which makes it challenging to find a feasible control strategy. To avoid this effect, ref. [9] proposes to use an ancillary control policy that feeds back the uncertainty of the predicted state through a suitably chosen gain matrix $\mathcal{K}$. Formally, this means that the control input is parameterized in the following form:

$$u(i|k) = \mu_\mathrm{u}(i|k) - \mathcal{K}\left(\mu_\mathrm{x}(i|k)\right)\left(x(i|k) - \mu_\mathrm{x}(i|k)\right). \tag{18}$$

where $\mu_\mathrm{x}(i|k)$ is the predicted mean of $x(i|k)$ and $\mu_\mathrm{u}(i|k) \in \mathbb{R}^{n_\mathrm{u}}$ is a deterministic input, corresponding to the optimization variables $\bar{\mu}_{\mathrm{u},k} = \mathrm{vec}(\mu_\mathrm{u}(0|k), \dots, \mu_\mathrm{u}(N_\mathrm{p}-1|k))$ in the minimization of $\mathbb{E}(J_k)$. As (18) indicates, $\mathcal{K}$ can be chosen to depend on $\mu_\mathrm{x}(i|k)$; however, for the sake of simplicity,[1] we will use a constant $\mathcal{K}$ in the sequel. More details about the properties and the selection of the gain matrix can be found in [9]. A particular case is discussed in Section 5.

Based on these considerations, the resulting optimal stochastic predictive control problem at time k can be formulated as follows:

$$\min_{\bar{\mu}_\mathrm{u}(k)} \mathbb{E}(J_k) \quad \text{subject to (6), (12), (16), (17), (18).} \tag{19}$$

Note that problem (19) is a highly complex problem, due to the non-linear propagation of the distribution of $w = \mathrm{vec}(x, u)$ via (6). Using the approximative prediction model derived in Section 2.4, the optimization problem (19) can be significantly simplified. Based on (11) and (12), $\mathbb{E}(J_k)$ can be approximated as

$$J_k^\mathrm{GP} = \sum_{i=1}^{N_\mathrm{p}} \|r(k+i) - C\mu_\mathrm{x}(i|k)\|_Q^2 + \sum_{i=0}^{N_\mathrm{p}-1} \|\mu_\mathrm{u}(i|k)\|_R^2 +$$

$$\sum_{i=1}^{N_\mathrm{p}} \mathrm{Tr}(Q\,C\,\Sigma_\mathrm{x}(i|k)\,C^\top) + \sum_{i=0}^{N_\mathrm{p}-1} \mathrm{Tr}(R\,\mathcal{K}\,\Sigma_\mathrm{x}(i|k)\,\mathcal{K}^\top). \tag{20}$$

The derivation of (20) is given in Appendix A.1. Furthermore, assume that $\mathcal{X}$ and $\mathcal{U}$ are polytopes, described by intersections of n_X and n_U number of half-spaces:

$$\mathcal{X} = \bigcap_{j=1}^{n_\mathrm{X}} \left\{x \mid S_{\mathrm{x},j}^\top x \leq s_{\mathrm{x},j}\right\},\quad \mathcal{U} = \bigcap_{j=1}^{n_\mathrm{U}} \left\{u \mid S_{\mathrm{u},j}^\top u \leq s_{\mathrm{u},j}\right\}.$$

Then, the following deterministic constraints

$$\mu_\mathrm{x}(i|k) \in \overbrace{\bigcap_{j=1}^{n_\mathrm{X}} \left\{x \mid S_{\mathrm{x},j}^\top x \leq s_{\mathrm{x},j} - \tau_{\mathrm{x},j}\left(\Sigma_\mathrm{x}(i|k)\right)\right\}}^{\mathcal{X}_\mathrm{t}(\Sigma_\mathrm{x}(i|k))} \tag{21a}$$

$$\mu_\mathrm{u}(i|k) \in \underbrace{\bigcap_{j=1}^{n_\mathrm{U}} \left\{u \mid S_{\mathrm{u},j}^\top u \leq s_{\mathrm{u},j} - \tau_{\mathrm{u},j}\left(\Sigma_\mathrm{x}(i|k)\right)\right\}}_{\mathcal{U}_\mathrm{t}(\Sigma_\mathrm{x}(i|k))} \tag{21b}$$

with $\tau_{\mathrm{x},j}(\cdot) = \pi\left(\frac{1-p_\mathrm{x}}{n_\mathrm{X}}, S_{\mathrm{x},j}, \cdot\right)$, $\tau_{\mathrm{u},j}(\cdot) = \pi\left(\frac{1-p_\mathrm{u}}{n_\mathrm{U}}, \mathcal{K}^\top S_{\mathrm{u},j}, \cdot\right)$, and $\pi(q, S, \Sigma) = \Phi^{-1}(1-q)\left(S^\top \Sigma\, S\right)^{\frac{1}{2}}$ imply the state and input constraints in (16), where Φ denotes the distribution function of the Gaussian distribution (see [25] for details). Finally, the deterministic *non-linear model predictive control* (NMPC) problem approximating (19) at time k can be formulated as follows:

$$\min_{\bar{\mu}_{\mathrm{u},k}} J_k^\mathrm{GP} \quad \text{subject to (11), (12), (17), (18), (21).} \tag{22}$$

The optimization problem (22) is also often called a GP-MPC problem, which is still difficult to solve, because (12) is highly non-linear in the decision variables $\bar{\mu}_\mathrm{u}$. The complexity is especially high, when the exact moments (7) approximation is used. In the next section, we propose an iterative procedure that can efficiently solve this problem.

[1] Including a $\mu_\mathrm{x}(i|k)$ dependence in $\mathcal{K}$ would result in a modified version of (20) and (21) which are rather straightforward to derive, but would increase the complexity of the resulting optimization problem significantly.

4 | EFFICIENT SOLUTION BY LPV ITERATIONS

To efficiently solve the optimization problem (22), we follow the concept of *linear parameter-varying* (LPV) predictive control of non-linear systems [23, 24]. First, the dynamic equations of the prediction model are rewritten in an LPV form by selecting certain state- and input-dependent terms as scheduling variables. By considering a given scheduling trajectory, the approximative prediction model and the GP-MPC problem (22) reduces to a simple linear time-varying predictive control problem that can be rapidly solved by *quadratic programming* (QP). After solving the QP, the resulting input sequence is then applied to simulate the approximative non-linear predictive model and based on the resulting state response, the scheduling is updated and the previous steps are repeated. The iteration continues as long as the scheduling updates converge.

4.1 | LPV reformulation

To formulate the LPV-MPC problem, we need to rewrite (12) in an LPV form. We propose the following LPV reformulation of (12):

$$\mu_{\rm x}(i+1|k) = A\left(p(i|k)\right)\begin{bmatrix}\mu_{\rm x}(i|k)\\ \mu_{\rm u}(i|k)\end{bmatrix} + c\left(p(i|k)\right), \tag{23}$$

where the scheduling vector is defined as

$$p(i|k) = \mathrm{vec}(\mu_{\rm x}(i|k), \mu_{\rm u}(i|k), \Sigma_{\rm x}(i|k)). \tag{24}$$

Note that (24) expresses the *scheduling map*, that is, how p relates to the original variables of (12), while, from the LPV point of view, it is considered as an independent, external variable. This particular choice turns (12) into a simple affine form where

(P.1) the non-linear dependency over $\mu_{\rm x}(i|k), \mu_{\rm u}(i|k), \Sigma_{\rm x}(i|k)$ is covered by the parameter-varying terms A and c;
(P.2) the variance transition (12b) vanishes as it is fully expressed by the trajectory p.

In terms of Property P.2, the variance dynamics (12b) are not included in the LPV prediction model (23), hence

(P.3) the cost terms in (20) can also be dropped as they become a sequence fixed to a chosen trajectory of p:

$$J_k^{\rm LPV} = \sum_{i=1}^{N_{\rm p}} \|r(k+i) - C\mu_{\rm x}(i|k)\|_Q^2 + \sum_{i=0}^{N_{\rm p}-1} \|\mu_{\rm u}(i|k)\|_R^2. \tag{25}$$

To complete our LPV reformulation, we can turn the non-linear state and input chance constraints (21) to

(P.4) a scheduling-dependent constraint set:

$$\mu_{\rm x}(i|k) \in \mathcal{X}_{\rm t}(p(i|k)), \quad \mu_{\rm u}(i|k) \in \mathcal{U}_{\rm t}(p(i|k)). \tag{26}$$

Note that this reformulation results in an LPV form with the least level of complexity, but considerable conservativeness as, in the minimization of (25), the resulting state variance $\Sigma_{\rm x}(i|k)$ does not play any role. Alternatively, it is possible to factorize c in (12a) in the form of $\breve{c}(p(i|k))\mathrm{vec}(\Sigma_{\rm w}(i|k))$ and the right-hand side of (12b) in terms of $\mathcal{A}_\mu(p(i|k))\mu_{\rm w}(i|k) + \mathcal{A}_\Sigma(p(i|k))\mathrm{vec}(\Sigma_{\rm w}(i|k))$ based on the approach given in [26] or in [27, Sec. 7]. This introduces an LPV state propagation for $\Sigma_{\rm w}(i|k)$, dropping Property P.2 and preserving the variance-dependent costs in (20), removing Property P.3. While this decreases conservatives, it increases complexity of the LPV form and the associated optimization problem.

4.2 | The iterative LPV-MPC algorithm

Based on the LPV reformulation discussed in Section 4.1, the NMPC problem (22) can be rewritten to an LPV-MPC problem as follows:

$$\min_{\bar{\mu}_{{\rm u},k}} J_k^{\rm LPV} \quad \text{subject to (17), (23), and (26),} \tag{27}$$

where $p(i|k) \in \mathrm{vec}(\mathbb{R}^{n_{\rm x}+n_{\rm u}} \times \mathbb{S}^{n_{\rm x}})$ with $i \in \mathbb{I}_0^{N_{\rm p}-1}$ is a given (arbitrary) scheduling sequence. Note that (27) is a simple QP that can be solved efficiently.

At time moment $k \in \mathbb{Z}$, let $\{\mu_{\rm u}^*(i|k)\}_{i=0}^{N_{\rm p}-1}$ be the optimal solution of the GP-MPC problem (22) and let $\mu_{\rm x}^*(i|k), \mu_{\rm u}^*(i|k), \Sigma_{\rm x}^*(i|k$ be the corresponding optimal state trajectory. Then, under the scheduling trajectory $p^*(i|k) = \mathrm{vec}(\mu_{\rm x}^*(i|k), \mu_{\rm u}^*(i|k), \Sigma_{\rm x}^*(i|k))$, the optimal solution of the GP-MPC problem (22) is a feasible solution of problem (27). For any other scheduling trajectory, (27) qualifies as an approximation of (22): if the solution of (27) corresponds to the given scheduling trajectory, then it is a solution of the GP-MPC problem (22) as well. Hence, to obtain the optimal solution of (22), the LPV-MPC problem (27) is solved iteratively with $\{p(i|k)\}_{k=0}^{N_{\rm p}-1}$ initialized by the trajectory from step $k-1$ or at $k=0$ is determined by a specific initialization scheme. In every iteration, (27) recast as a simple QP and solved by any standard software, then, based on the resulting input sequence, the scheduling is re-computed according to the non-linear state recursion (12). The resulting iterative MPC scheme is outlined in Algorithm 1. The specific steps are detailed in the sequel.

Initialization: When the predictive controller is started at $k=0$ the scheduling sequence $\{p(i|0)\}_{i=0}^{N_{\rm p}-1}$ is required to be determined to start solving the first QP (27). The following initialization scheme is proposed (Line 1):

$$p(i|0) = \mathrm{vec}(\mu_{\rm x}(0), 0_{1\times n_{\rm u}}, \Sigma_{{\rm x},0}), \quad \forall i \in \mathbb{I}_0^{N_{\rm p}-1}, \tag{28}$$

where $\mu_{\rm x}(0) = x(0)$ is the first measured state with a point distribution ($\Sigma_{{\rm x},0} = 0$).

ALGORITHM 1 Iterated LPV-MPC solution of GP-MPC.

```
1:  initialization: set $k \leftarrow 0$, measure $x(0)$ and set $\mu_x(0) = x(0)$ and
        $\Sigma_{x,0} = 0$, and set $p(i|0)$ with $i \in \mathbb{I}_0^{N_p-1}$ via (28)
2:  loop
3:      repeat
4:          solve (27) to obtain $\mu_u(i|k)$
5:          simulate (12) with $\mu_u(i|k)$ to obtain $\mu_x(i+1|k)$, $\Sigma_x(i+1|k)$
6:          update $p(i|k)$ using $\mu_u(i|k)$, $\mu_x(i|k)$, $\Sigma_x(i|k)$ according to (31)
7:      until $p(i|k)$ has converged or $n_{loop}$ iterations reached
8:      apply $u(k) = \mu_u(0|k)$
9:      $k \leftarrow k+1$
10:     measure the actual state $x(k)$ according to (1)
11:     update $p(i|k)$ by (24) and set $\mu_x(k) = x(k)$
12: end loop
```

Solving the LPV-MPC problem: The LPV-MPC problem (27) along the fixed scheduling trajectory $p(i|k)$ provides a fixed sequence of variation of the coefficient matrices $A(p(i|k))$ and $c(p(i|k))$ of (23) and the scheduling-dependent polytopes $\mathcal{X}_t(p(i|k))$ and $\mathcal{U}_t(p(i|k))$ in (26). Along this variation, the problem is recast and solved as a standard QP problem. (Line 4)

Simulation of the NL prediction model: The non-linear approximative model (12) driven by the control policy (18) is used to compute the resulting state variation in terms of $\mu_x(i|k)$ and $\Sigma_x(i|k)$ for the found optimal inputs $\mu_u(i|k)$ in the previous step of the LPV-MPC scheme. (Line 5)

Updating the scheduling sequence: $p(i|k)$ is updated with the computed inputs $\mu_u(i|k)$ and the simulated states $\mu_x(i|k)$, $\Sigma_x(i|k)$ according to (24) in Line 6.

Measure of convergence: The iterative optimization of the LPV-MPC problem (27) is continued until the computed scheduling trajectory (24) converges. The convergence can be expressed in any norm of the corresponding vectors. Here, we propose to use an ℓ_∞ norm–based convergence condition:

$$\max_{i \in \mathbb{I}_0^{N_p-1}} \| p(i|k) - \mathrm{vec}(\mu_x(i|k), \mu_u(i|k), \Sigma_x(i|k)) \|_\infty < \varepsilon_p, \tag{29}$$

where $\varepsilon_p > 0$ is a small threshold value. If convergence does not happen in $n_{loop} \in \mathbb{Z}_+$ number of iterations, then the execution is stopped to limit the computational budget of the LPV-MPC scheme.

Applying the input: After the LPV-MPC computations are terminated, the first value of the computed mean input sequence $\mu_u(0,k)$ is applied to the system according to the receding horizon principle. Note that $\mathcal{K}$ does effect this input as, due to the used measurement model (17), $x(0|k) \sim \mathcal{N}(x(k), 0)$. In fact, $\mathcal{K}$ plays a role in shaping the GP-MPC control performance as it influences the actual future planning of the input sequence in view of the uncertainty of the state predictions, that is, it realizes a default feedback action on the entire state distribution. Hence, an aggressive $\mathcal{K}$ results in an MPC law that tries to suppress deviations from the predicted mean at the expanse of more input power while a low gain $\mathcal{K}$ allows state uncertainty where mainly the mean is governed by the input actions. (Line 8)

Measuring the new state response: When the response of the system to the implemented control action is measured, it is assumed that the measurement is exact, allowing to set $x(0|k) \sim \mathcal{N}(x(k), 0)$ for the next control cycle.

Propagation of the scheduling sequence: When a new control cycle starts by advancing the time k, initialization of the scheduling sequence $p(i|k)$ is required to be accomplished for starting the LPV-MPC iterations. The previously converged input sequence $\mu_u(0|k-1)$, from step $k-1$, is used to initialize the initial scheduling sequence for starting the LPV-MPC iterations:

$$\mu_u(i|k) = \mu_u(i+1|k-1), \quad \forall i \in \mathbb{I}_1^{N_p-2}. \tag{30a}$$

Note that $\mu_u(N_p|k-1)$ has not been computed in the previous step, hence to support recursive feasibility

$$\mu_u(N_p - 1|k) = \mu_u(N_p - 2|k) \tag{30b}$$

is taken. Additionally, due to the current state measurement $x(k)$, more accurate information on $\mu_x(0|k)$ is available than the predicted sequence of state variation in step $k-1$. Hence using $x(0|k) \sim \mathcal{N}(x(k), 0)$ and the propagated input sequence $\mu_u(i|k)$, the non-linear approximative model (12) is used to compute a new initial state variation in terms of $\mu_x(i|k)$ and $\Sigma_x(i|k)$. Then, the initial $p(i|k)$ is computed according to (24).

Remark 1. Convergence of the LPV-MPC scheme can be accelerated if in early iterations the scheduling sequence $p(i|k)$ is updated with the state mean sequence $\hat{\mu}_x(i|k)$ computed in the solution of the LPV-MPC problem (27) in Line 4 rather than the simulated state mean sequence $\mu_x(i|k)$ obtained in Line 5, namely,

$$p(i|k) = \mathrm{vec}(\hat{\mu}_x(i|k), \mu_u(i|k), \Sigma_x(i|k)). \tag{31}$$

While the simulated mean μ_x corresponds to the true predicted state variation for the optimized input sequence, $\hat{\mu}_x$ provides an LPV conversion of the next optimization problem along the solution of the previous one, often providing better convergence rate when $p(i|k)$ is far from the optimal solution of the GP-MPC problem. This update is suggested till the LPV-MPC solution drastically changes between iterations, that is, $\hat{\mu}_x$ and μ_x have significant differences. A proposed way to test this is in terms of

$$\|\mu_x - \hat{\mu}_x\|^2_{\ell_\infty} = \max_{i \in \mathbb{I}_1^{N_p}} \|\mu_x(i|k) - \hat{\mu}_x(i|k)\|_2^2 < \varepsilon_{acc}, \tag{32}$$

where $\|\cdot\|_2$ denotes the Euclidean norm and ε_{acc} is a sufficiently large threshold value, typically 10^{-3}. Note that as the main computational load comes from iterating the LPV-MPC solution, especially in terms of simulation of the non-linear prediction model, faster convergence has a direct influence on the achievable control cycle time with GP-MPC. The maximum number of acceleration steps can be also maximized in terms of $n_{\mathrm{acc}} \in [5, 10]$ with $n_{\mathrm{acc}} \leq n_{\mathrm{loop}}$.

Remark 2. A further possible acceleration of the proposed scheme is to skip the iterations in the LPV-MPC problem, that is, to set $n_{\mathrm{loop}} = 1$ for Line 7. In this way, the receding-horizon principle is used to converge to the set point and to the optimal scheduling sequence $p^*(i|k)$ simultaneously. While there is little knowledge about the convergence conditions of the overall scheme, a similar concept has been used successfully in applications of deterministic LPV predictive control [23].

Remark 3. In this paper, our objective is to develop an efficient solution of the GP-MPC problem (22). Ensuring recursive feasibility and overall stability of the original GP-MPC scheme, for example, by the choice of $\mathcal{K}$ in (18) and terminal ingredients, are rather important for a safe operation of such controllers, but from the authors' knowledge these are open problems and are out of the scope of the current work.

5 | EXAMPLES

5.1 | Robotic arm simulation example

As an illustrative example, we consider motion control of the simulation model of the 4DOF QArm robotic arm by Quanser. The actual robotic arm and its skeleton model is illustrated in Figure 1.

5.1.1 | Dynamic motion model

Under ideal conditions, the equations of motion of the arm can be expressed in the form of a continuous-time Euler–Lagrangian model:

$$H(q)\,\ddot{q} + h(q, \dot{q}) = \tau, \tag{33}$$

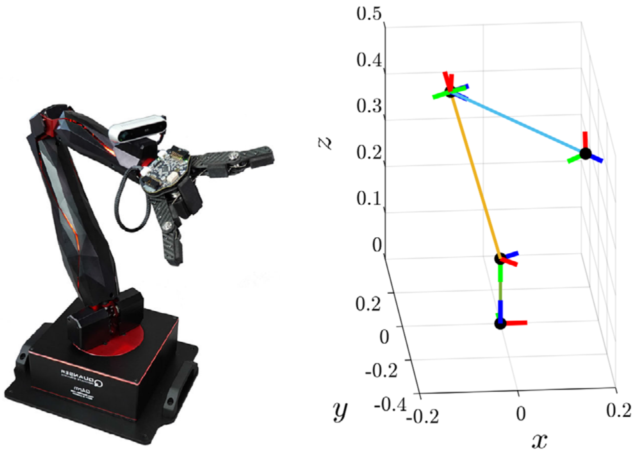


FIGURE 1 4DoF QArm robotic arm by Quanser: actual system (left panel) and its skeleton model (right panel).

where $q : \mathbb{R} \to \mathbb{R}^{n_{\mathrm{q}}}$ represents the four joint angles ($n_{\mathrm{q}} = 4$), namely, base yaw, shoulder pitch, elbow pitch, and wrist yaw, while $\tau(t) \in \mathbb{R}^{n_{\mathrm{q}}}$ denotes the corresponding torques provided by servo motors at these joints at time moment $t \in \mathbb{R}$. In (33), $H(q)$ denotes the inertia matrix, whereas $h(q, \dot{q})$ comprises the Coriolis, centrifugal, and gravitational terms.

With respect to the real robot arm, the model (33) only expresses the ideal motion dynamics while the actual arm can experience further dynamic effects such as friction, unbalanced construction, further loads at the end effector, and so on. Assume that the actual dynamics can be described in the form of

$$H_{\mathrm{o}}(q)\,\ddot{q} + h_{\mathrm{o}}(q, \dot{q}) = \tau, \tag{34}$$

where H_{o} and h_{o} are considered to be unknown.

In practice, torque-controlled robotic arms are often regulated by using the concept of *computed torque control*, that is, based on the assumed model (33), τ is chosen as

$$\tau = H(q)\,u + h(q, \dot{q}), \tag{35}$$

where $u : \mathbb{R} \to \mathbb{R}^{n_{\mathrm{q}}}$ corresponds to virtual inputs. When (35) is applied to the actual system (34), the dynamics of motion become

$$\ddot{q} = u + \underbrace{H_{\mathrm{o}}^{-1}(q)\left(\left(H(q) - H_{\mathrm{o}}(q)\right) u + h(q, \dot{q}) - h_{\mathrm{o}}(q, \dot{q})\right)}_{g_{\mathrm{d}}(x,u)}, \tag{36}$$

where $x = \mathrm{vec}(q, \dot{q})$, which can be rewritten as

$$\dot{x} = \underbrace{\begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix}}_{A} x + \underbrace{\begin{bmatrix} 0 \\ I \end{bmatrix}}_{B} \left(u + g_{\mathrm{d}}(x, u)\right). \tag{37}$$

If the assumed ideal model (33) matches the actual dynamics (34), that is, $g_{\mathrm{d}}(x, u) = 0$, (37) reduces to four decoupled second-order integrators that are easy to control. However, this almost never happens in practice and the presence of $g_{\mathrm{d}}(x, u)$ deteriorates the control performance.

For the application of our methodology to estimate g_{d} and compensate its effect by GP-MPC, (37) is written in discrete time:

$$x^+ = \Phi\, x + \Gamma\, \left(u + g_{\mathrm{d}}(x, u)\right), \tag{38}$$

with $\begin{bmatrix} \Phi & \Gamma \\ 0 & I \end{bmatrix} = \exp\left(T_{\mathrm{s}} \begin{bmatrix} A & B \\ 0 & 0 \end{bmatrix}\right)$ corresponding to complete *zero-order hold* (ZOH) discretization with sampling period $T_{\mathrm{s}} = 0.1$ (s) applied to the integrator model $\dot{x} = Ax + Bu$.

In this simulation study, the actual dynamics (34) and the assumed model (33) are considered to differ in the moments of inertia, the mass of each link, the position of the centre of gravity of each link, and the mass of the payload at the end effector. To quantify how much H and h differ from H_{o} and h_{o}, the following relative distances are computed:

$$\max_{q} \frac{\|H(q) - H_{\mathrm{o}}(q)\|_{\mathrm{F}}}{\|H(q)\|_{\mathrm{F}}} \approx 0.2249, \tag{39a}$$

$$\max_{q,\dot{q}} \frac{\|h(q,\dot{q}) - h_{\mathrm{o}}(q,\dot{q})\|_2}{\|h(q,\dot{q})\|_2} \approx 0.2584, \tag{39b}$$

where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm. As seen above, the worst-case difference between the functions is about 22–25%.

As the inverse dynamics is not perfectly known, the term $g_{\mathrm{d}}(x,u)$ is relevant and affects directly the velocity dynamics. However, we assume that the additive model error corresponding to the wrist yaw dynamics (last coordinate of vector $g_{\mathrm{d}}(x,u) \in \mathbb{R}^4$) is negligible in this case study. Moreover, functions H and h of the hypothetical model (33) are independent of the base yaw (q_1) and the wrist yaw (q_4), therefore, the unknown term $g_{\mathrm{d}}(x,u)$ is modelled by three GPs:

$$g_{\mathrm{d}}(x,u) \approx \mathcal{GP}(w) = \mathrm{vec}(\mathcal{GP}_{\mathrm{r}}(\tilde{w}), 0), \tag{40}$$

with $\mathcal{GP}_{\mathrm{r}} : \mathbb{R}^{10} \to \mathbb{R}^3$, where the essential arguments of the GPs are $\tilde{w} = \mathrm{vec}(q_2, q_3, \dot{q}, u)$.

As proposed in [6], a GP-based MPC controller is applied to (37), which employs the following prediction model:

$$\hat{x}^{+} = \Phi x + \Gamma\left(u + \mathcal{GP}(w)\right) = \begin{bmatrix}\Phi & \Gamma\end{bmatrix} w + \Gamma\,\mathcal{GP}(w), \tag{41}$$

where the unknown term $g_{\mathrm{d}}(x,u)$ is replaced by a GP as described in (40). Although the controller is designed and operated in discrete time, the computed input is applied to the deterministic CT model (36) with zero-order hold during the sampling period of length T_{s}.

5.1.2 | Data generation

To capture the unknown dynamics and the discretization error, the measurement data is collected during a continuous-time simulation of (34), such that the state is measured at sampling rate T_{s} and the input is applied through a synchronized zero-order hold. When the inverse dynamics is not precisely known, the feedback linearization-based control does not work efficiently. Therefore, to provide a rough trajectory tracking for the partly known system model (34) in the joint space, we applied the feed-forward non-linear control approach presented in [28, Sec. III.A], which is based on (35) computed along the reference instead of q. Then, the inverse of the linearizing feedback (35) allows us to compute the acceleration input u, which would have produced the same torque via (35) as the applied control scheme. Finally, the measured reduced GP outputs are prepared as follows: $\tilde{z}(j) = \Gamma_{\mathrm{g}}^{\dagger}\,(x^{+}(j) - \Phi\, x(j) - \Gamma\, u(j))$, where $j = \mathbb{I}_1^N$ and $\Gamma_{\mathrm{g}}^{\dagger}$ is the left inverse of $\Gamma_{\mathrm{g}} = \Gamma \cdot \begin{bmatrix} I_{3\times 3} \\ 0 \end{bmatrix} \in \mathbb{R}^{8\times 3}$. Additional noise is not considered in this study, as the main focus of the paper is to solve GP-MPC problems efficiently rather than to analyze the robustness of the GP-MPC scheme against noise.

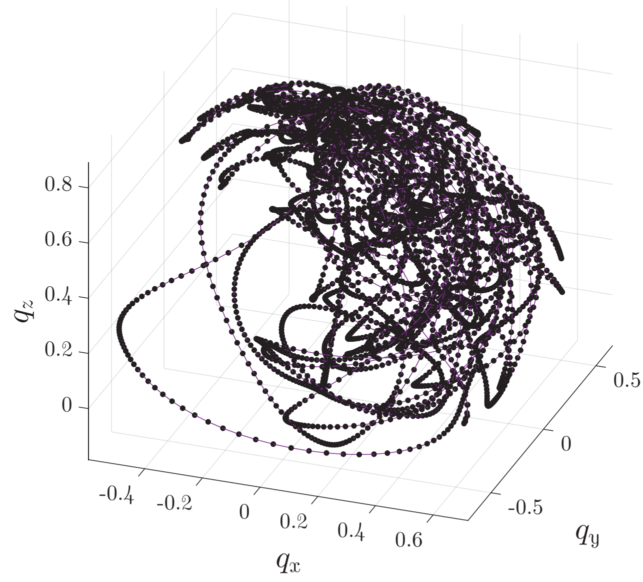


FIGURE 2 Reference motion trajectory in the world space followed by the end-effector of the simulated arm during collection of the training data. The black dots illustrate the place where a training point was recorded with sampling time T_{s}.

For data generation, the reference trajectory has been designed as a spline connecting 121 distinct points in the joint space. The trajectory with a length 500 (s) has been optimized such that the joint positions, velocities, and accelerations meet the limitations prescribed by the manufacturer. The trajectory followed by the simulated plant and the $N = 5000$ recorded training points are illustrated in the world space in Figure 2. Let $\mathcal{D}_N$ denote the set of all collected training points available for GP estimation.

5.1.3 | GP estimation

Using $\mathcal{D}_N$, the three GPs in $\mathcal{GP}_{\mathrm{r}}$ (labelled by $i = 1, 2, 3$) have been estimated independently in two steps. First, the hyperparameters θ_i have been optimized by maximizing the marginal likelihood of $\tilde{z}_i$ under θ_i and the observed $\mathcal{D}_N$ [2, Sec. 5.4.1]. Then, inspired by [7], a reduced, sparse data dictionary $\mathcal{D}_{N_{\mathrm{s},i}} \subset \mathcal{D}_N$ of $N_{\mathrm{s},i}$ samples are formulated, to decrease the dimensionality of the GP regression. The dictionary entries are selected iteratively, from the original data set $\mathcal{D}_N$ based on the maximum predictive variance of the samples conditioned on the actual dictionary. The collection of entries to the dictionary is terminated, when appending further training points would not reduce the predictive variance significantly. In this way, the three (independent) dictionaries are filled with $N_{\mathrm{s},1} = 200$, $N_{\mathrm{s},2} = 30$, and $N_{\mathrm{s},3} = 100$ samples, respectively.

5.1.4 | Predictive control designs

To evaluate the performance of the proposed approach four control schemes are compared:

LTI-Oracle MPC: When the inverse dynamics of the actual system (34) are perfectly known, the non-linear components can be completely eliminated from the feedback linearized loop behaviour. The remaining integrator dynamics $\dot{x} = Ax + Bu$ in discrete time can be controlled by a simple LTI-MPC as follows:

$$\min_{u(0|k)} J_k \quad \text{subject to } x(0|k) = x(k), \tag{42a}$$

$$x(i+1|k) = \Phi\, x(i|k) + \Gamma\, u(i|k), \tag{42b}$$

$$u(i|k) \in \mathcal{U}. \tag{42c}$$

The resulting $u(0|k)$ is then applied on (38) which is considered to be the simulation model describing the robot dynamics in discrete time. We call the (42)-based predictive controller an LTI-Oracle design as it is based on an LTI prediction model and perfect computed torque control $\tau = H_{\mathrm{o}}(q)\, u + h_{\mathrm{o}}(q, \dot{q})$ corresponding to $g_{\mathrm{d}} = 0$ in (38). Note that the LTI-Oracle controller is not a realistic approach as it presumes the perfect knowledge of the partly unknown system model (34) and it represents the best achievable performance by any predictive scheme under the considered J cost function.

LTI-MPC: The same LTI-MPC (42) is applied to (38), when the inverse dynamics are not perfectly known, that is, $g_{\mathrm{d}} \neq 0$.
NL-Taylor MPC: Based on the obtained GP model of g_{d}, the GP-MPC approach as proposed by [25] is applied to control (38). In each sampling period, we solve the non-linear MPC problem (22), where the moments of the GP in the prediction model (12) are approximated using a first-order Taylor approximation as presented in (13a). In the NMPC computations, the control input policy is determined in the feedback form (18), where the constant feedback gain $\mathcal{K}$ is computed for the ideal integrator model using a classical LQR design with weight matrices:

$$\begin{aligned} Q_{\mathrm{LQR}} &= \mathrm{diag}(1, 10, 100, 0.1, 1, 1, 1, 1), \\ R_{\mathrm{LQR}} &= I_4. \end{aligned} \tag{43}$$

The weights in Q_{LQR} balance the predicted uncertainties of the four joint positions and angular velocities.
LPV-EM MPC: Finally, we apply the proposed GP-MPC approach to (38) using a prediction model (12) where the moments of the GP are computed analytically according to (14a). In the feedback input policy (18), we considered the same gain matrix K, obtained for the NL-Taylor approach. The iterated control computations were processed as detailed in Algorithm 1, such that the convergence threshold in (29) has been set to $\varepsilon_{\mathrm{p}} = 10^{-3}$, while the maximum number of QP iterations has been $n_{\mathrm{loop}} = 20$. Algorithm 1 has been executed with the acceleration approach described in Remark 1 with $n_{\mathrm{acc}} = 10$ and $\varepsilon_{\mathrm{acc}} = 10^{-3}$.

Cost function. In the four MPC approaches, three different cost functions appear. The LTI approaches consider J_k introduced in (15) in the deterministic sense as there is no uncertain stochastic element in the prediction model. The NL-Taylor method considers the expected cost J_k^{GP} in (20), while the LPV-EM works with the simplified form J_k^{LPV} of this costs function in terms of (25). In all MPC approaches, the same weight matrices have been used:

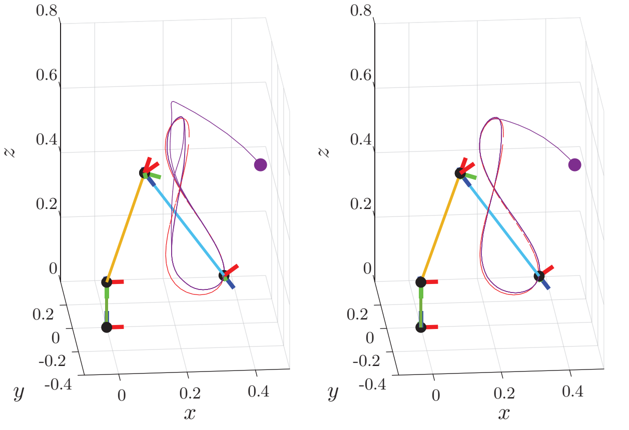


FIGURE 3 Reference and output trajectories in the world space of the QArm system controlled by the NL-Taylor (NMPC (22) with Taylor approximated moments (13a)) (left), and the LPV-EM (sequential LPV-MPC design (Algorithm 1) with analytical moment calculations) (right) GP-MPC methods. The dashed line illustrates the periodic reference trajectory of the end effector, whereas, the solid line is the actual trajectory of the effector with the initial position illustrated by a solid dot.

$$Q = 10\, I_{4\times4}, \quad R = 10^{-2} I_{4\times4}. \tag{44}$$

Input constraints. The feasible domain for the input $u \in \mathcal{U}$ is fixed as $\mathcal{U} = [-10, 10]^4$. This domain ensures the computed torque (35) to meet the manufacturer's recommended limitations at each admissible state. In the NL-Taylor and LPV-EM approaches, the tightened input constraints in (21) and (26), respectively, are enforced with a $p_{\mathrm{u}} = 95.45\%$ probability level. In this illustrative example, state constraints are not prescribed, as the reference tracking ensures a feasible solution in itself.

Further common parameters. In each case, the controller execution was simulated for 17.5 seconds ($k \in \{0, \ldots, 174\}$) with a periodic reference trajectory. The prediction horizon of the four predictive controllers has been set to $N_{\mathrm{p}} = 50$.

Simulation environment. The four control approaches are applied to the continuous-time system (36) with ZOH actuation and synchronized sampling with T_{s}. The initial state in each run is considered to be in a fixed rest position, displayed in Figure 3.

The computations have been performed in MATLAB on a desktop PC with Intel Core i5-4590 CPU at 3.3 GHz and 32 GB of RAM. For algorithmic differentiation, we used CasADi [29]. To solve non-linear optimization problems, we used IPOPT [30], an interior point line search algorithm, with the MUltifrontal Massively Parallel sparse direct Solver (MUMPS) [31, 32]. To solve quadratic problems, we used MOSEK [33].

TABLE 1 Performance comparison of the proposed approach.

	LTI-Oracle	LTI	NL-Taylor	LPV-EM	LPV-EM (accelerated)
1. Tracking error$\|\|y-r\|\|_{\ell\infty}$	0.0033	0.3229	0.0931	0.0226	0.0226
2. Offline model construction (s)	3	3	245	3	3
3. Avg. solver time (s)	–	–	5.1	0.0142	0.0124
4. Avg. simulation time with EMM (s)	–	–	–	0.7760	0.7028
5. Avg. number of iterations in a control cycle	–	–	1	11	6
6. Avg. control cycle time (s)	0.04	0.04	5.1	8.6777	4.29

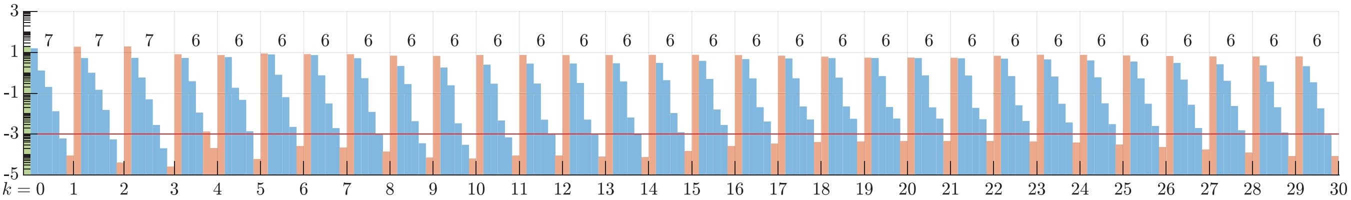


FIGURE 4 Scheduling difference during the LPV-MPC iterations with the acceleration option (Remark 1) and convergence threshold $\epsilon_p = 10^{-3}$.

5.1.5 | Results and discussion

With the discussed MPC schemes, (36) has been simulated in a closed loop w.r.t. the joint equivalent of the reference trajectory depicted in (3). The obtained responses have been analyzed based on the indicators listed below and the results are summarized in Table 1.

1. The tracking error is quantified in the ℓ_∞ sense:

$$\|r - Cx\|_{\ell_\infty} = \max_{k \geq k_0} \|r_k - Cx_k\|_2. \tag{45}$$

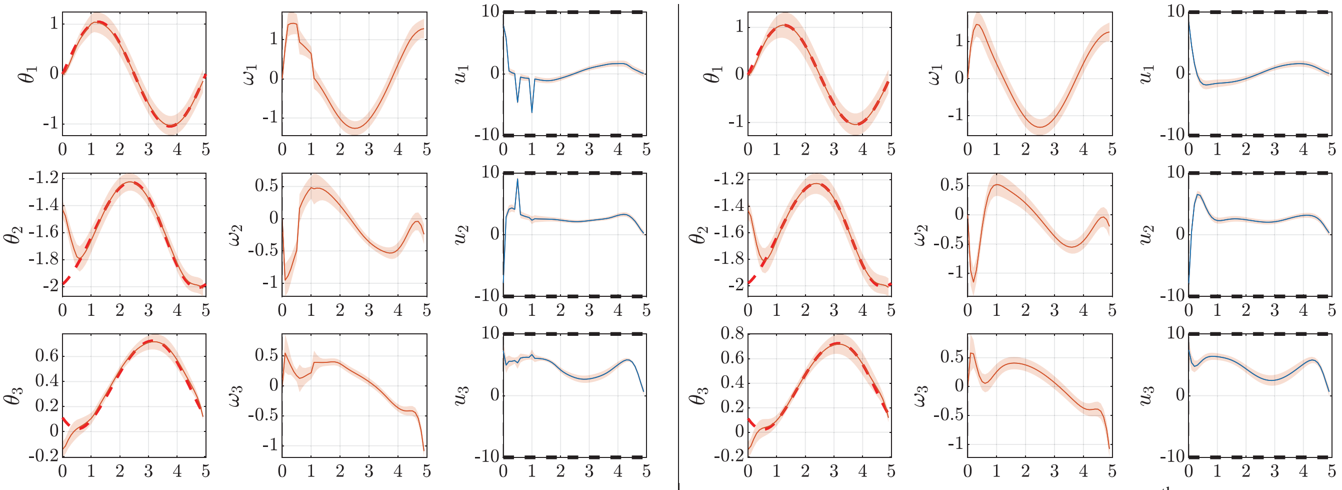
   As the reference trajectory starts from a different position than the initial pose of the arm, the tracking error in the first few steps $k \leq k_0 = 20$ (in the first 2 seconds) is a few orders of magnitude higher than the overall cumulative error after the transient behaviour. To avoid that this part dominates the quantification of the tracking error, the error measures are only computed for $k \geq k_0$.
2. We recorded the time required for offline construction of the optimization model. Note that this is only required to be done once before the control cycles start.
3. The average solver time (Step 4 in Algorithm 1) in each iteration over the 175 time steps of the entire simulation is reported in Table 1. Note that in one control cycle, the calculation of the-to-be applied input u_k to the system takes different number of iterations with the proposed LPV schemes.
4. The average simulation and scheduling calculation time (Steps 5 and 6 in Algorithm 1) in each iteration over the entire simulation is also provided for the introduced LPV schemes with EM-based propagation model.
5. The average number of iterations is also provided for each input calculation cycle (Steps 3–11 in Algorithm 1).
6. Finally, the average total time needed for each input calculation cycle is also reported in Table 1.

In Figure 3, the resulting end effector trajectories in the world space are depicted when the arm is controlled by the NL-Taylor and the LPV-EM predictive controllers. The predicted state and input trajectories computed at time $k = 0$ are illustrated in Figure 5 for both GP-based approaches.

In Figure 4, we present the time evolution of the $\log_{10}$ of the scheduling difference (29) during the iterative control computations of the LPV-EM method. The vertical grid lines, which bound the intervals $[k, k+1]$, $k \in \mathbb{I}_0^{30}$ illustrate the end of each sequence of iterations, after which the first input value is applied to the system at time k. The numbers at the top of the axes constitute the number of LPV-MPC iterations required in the corresponding sampling period. The red horizontal line highlights the scheduling convergence threshold ϵ_p. The orange and blue bars illustrate the difference in scheduling after a single MPC computation. The orange bars highlight the case when the previous *simulated* mean μ_x was used to update the scheduling, whereas, the blue bars correspond to the iterations, when the previous solution $\hat{\mu}_x$ of the LPV-MPC was used (31).

Relevance of the GP

The tracking error of the LTI approach, when the inverse dynamics is not perfectly known, is two orders of magnitude higher compared to the LTI-Oracle controller, where the inverse dynamics is presumed available. This fact allows us to conclude that the unknown dynamic component $g_d(x, u)$ in (38) is relevant and its impact to the overall dynamics is not negligible. Therefore, the augmentation of the prediction model (41) with an additive Gaussian process model is reasonable.



(a) Sate-prediction by **NL-Taylor**.

(b) Sate-prediction by **LPV-EM** after the 7th iteration.

FIGURE 5 State and input predictions at time $k = 0$ made by NL-Taylor in panel (A) and LPV-EM after the 7th iteration in panel (B). The dashed red line is the reference output r to track, the dashed black lines visualize the input constraints, the blue lines illustrate the computed states $\hat{\mu}_x$ and inputs μ_u, the orange lines are the simulated state expectations μ_x, the shaded orange areas illustrate the uncertainty of the simulated trajectories μ_x and the computed inputs μ_u. The shaded areas constitute the 95.45% confidence intervals of the marginal distributions of the corresponding random variables.

Initialization

The complexity of the non-linear optimization model built for NL-Taylor method is significantly higher compared to a QP required for the LPV-EM controller. Therefore, the model construction requires more time and hardware resources.

Precision

In this simulation example, the proposed LPV strategy with EM achieved a lower tracking error compared to the state-of-the-art approach (NL-Taylor). This can be explained by the fact that the analytically calculated moments (14a) provide a more accurate prediction with model (12) than the Taylor approximated moments (13a). Moreover, the time evolution of the scheduling error (Figure 4) suggests that the iterated solutions show a good convergence to an exact solution of the non-linear problem (22) equipped with exact moments. It is worth remarking that the iterated LPV-EM is potentially more conservative (overly cautious) compared to the NMPC with EM, since the variance cost terms of (20) are not considered during the QP iterations. However, the obtained predictions in Figure 5 suggest that the iterated LPV-EM approach is conservative to an acceptable degree compared to the NL-Taylor method. Namely, the uncertainty of the predictions is not significantly higher compared to non-linear method.

Computational efficiency

The computational time indicators in Table 1 show that the moment calculations in the LPV approach accounted for the bulk of the processing time during the control execution. However, the overall processing time is lower than the cumulative solver time of the NL-Taylor approach. Note that this is a significant achievement in itself, as even if NL-EM MPC is less conservative in its prediction model, it has been reported to have much higher computational cost than the NL-Taylor variant. The fact that the LPV-based solution makes the EM option faster than the NL-Taylor, brings higher accuracy reachable with even lower computational time compared to the existing state-of-the-art. This highlights the overall magnitude of the contributions of this paper.

Due to computational limitations, the comparison of the proposed approach with the NMPC (22) using analytically computed moments (NL-EM MPC) was not possible for this example. Therefore, in the following subsection, we consider an academic example to perform a relevant comparison.

5.2 | Academic example

Consider the forced Van der Pol oscillator with an additive non-linear term as follows:

$$\dot{x}_1 = x_2, \tag{46a}$$

$$\dot{x}_2 = \left(1 - x_1^2\right)x_2 - x_1 + u + g_d(x_2), \tag{46b}$$

where x_1 is the position, x_2 is the velocity of the oscillator, and u is an external force input. The output is the position x_1, which is meant to follow a reference trajectory. In (46), function g_d represents a non-linear friction phenomenon, which depends only on the velocity of the system. The friction function is illustrated in Figure 6 (solid black line). In Figure 8, the limit cycle of (46) is displayed with and without the friction term.

In this simulation study, the non-linear friction g_d is considered to be unknown and hence approximated by a GP. To construct a training set $\mathcal{D}_N = \left\{(x_2(j), z(j))\right\}_{j=1}^{N}$, we have generated a few, not identically distributed test points $x_2(j)$.

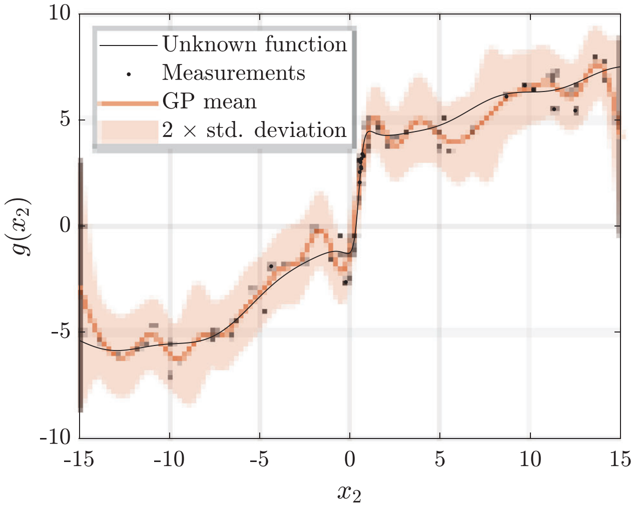


FIGURE 6 Non-linear friction characteristics and its Gaussian Process model for the forced Van der Pol oscillator.

Then, the corresponding training outputs $z(j) = g_d(x_2(j)) + v(j)$ have been polluted by an additive noise v with variance $\sigma_v^2 = 0.36$. The hyperparameters have been tuned by maximizing the marginal likelihood of z under the training samples $\mathcal{D}_N$. The data points are visible in Figure 6 by the black dots. The mean and variance of the estimated $\mathcal{GP}_r(x_2)$ are displayed in Figure 6 with a 95% confidence bound.

A discrete-time prediction model for (46) is determined using the explicit Euler discretization method:

$$\hat{x}^+ = f(w) + T_s\mathcal{GP}(w), \tag{47}$$

with sampling period $T_s = 0.02$ (s) and $w = \text{vec}(x, u)$. In (47), the nominal model is

$$f(w) = \begin{bmatrix} x_1 + T_s\, x_2 \\ x_2 + T_s\,(1 - x_1^2)\, x_2 - T_s\, x_1 + T_s\, u \end{bmatrix}, \tag{48}$$

whereas the unknown term $g_d(x_2)$ in (47) is simply replaced by the estimated $\mathcal{GP}(w) = \mathcal{GP}_r(x_2)$.

A deterministic approximation (12) for (47) is derived according to Sections 2.3 and 2.4 using exact moment matching (14a). We consider two control approaches.

NL-EM MPC: Corresponding to (46) controlled by the NL GP-MPC approach (22) with analytic moment calculations.
LPV-EM MPC: When the proposed iterated LPV approach described in Algorithm 1 is applied on (46). The convergence threshold in (29) is set to $\varepsilon_p = 10^{-3}$, and the maximum number of LPV-MPC iterations is $n_{\text{loop}} = 10$. Algorithm 1 is executed with the acceleration approach described in Remark 1 with $n_{\text{acc}} = 5$ and $\varepsilon_{\text{acc}} = 10^{-3}$.

In both MPCs, we considered the same weight matrices $Q = 10^4 I_{2\times2}$, $R = 0.1$ in the cost functions. The prediction horizon in both cases was $N_p = 50$, whereas the simulation horizon was $T = 14$ (s) (i.e., $k \in \{0, \dots, 699\}$). For simplicity, the input was optimized with $K \equiv 0$ in (18), and no input and state constraints have been considered in this example. The inputs computed for the stochastic DT model (47) have been applied to the deterministic CT model (46) under ZOH actuation and synchronized output sampling.

The resulting position and velocity trajectories in comparison with the reference and the applied inputs are shown in Figure 7 for both GP-MPC approaches. The state trajectory achieved by the NL GP-MPC approach is illustrated in the phase plot in Figure 8 by the yellow dotted line.

The maximal error (0.68) between the achieved position output and the reference is practically the same for the two controllers. Their relative difference is within 0.3%. In this simulation example, the proposed approach shows a good convergence to the non-linear problem and requires in average 4 ± 1 iterations . The average processing time of a single input calculation required by the LPV approach is 0.115 (s), whereas an input calculation during the non-linear MPC required an order of magnitude higher (1.12 (s)) average processing time.

The implementations are available in the public repository [34].

6 | CONCLUSION

In this paper, a novel LPV approach is proposed for a rapid solution of GP-MPC problems for discrete-time non-linear state-space models augmented with GPs. The core idea is that by converting the mean and covariance prediction model both under Taylor and exact moment (EM) matching-based approximation to an LPV form, the GP-MPC problem reduces to a quadratic program. By refining the scheduling via successive solutions of the QP, rapid convergence to an optimum of the GP-MPC problem can be observed, although there are no analytic convergence guarantees. Compared to the majority of the sequential convexification techniques, the presented approach has two main advantages. First, it does not require to compute or approximate the gradient of the dynamic equation, but only simulate the non-linear dynamics over the control horizon. This makes it possible to perform MPC computations with a more difficult prediction models, in particular, with exact GP moments. Secondly, the predicted state evolution is provided using simulations of the non-linear mean-variance prediction model.

Through illustrative examples, it has been demonstrated the proposed LPV approach is capable to outperform sequential convexification–based solution of the GP-MPC problem in terms of realizing EM based GP-MPC, when the non-linear solution is non-computable, and execute it with lower computational time than the much cheaper Taylor approximation based non-linear MPC solution without any performance loss. Further research is aimed at improving analytic computation of the moments which currently contribute the largest to the overall solution time of the proposed method.

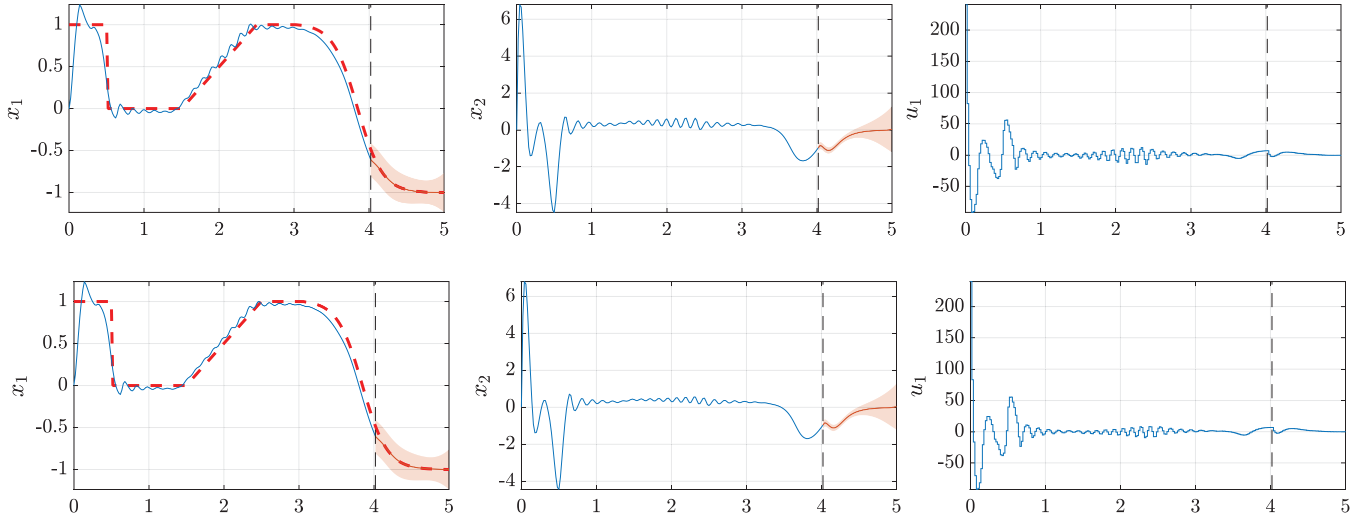


FIGURE 7 State and input trajectories (blue lines) achieved by LPV-EM (top) and NL-EM (bottom) GP-MPC methods compared to the reference trajectory (dashed red line). The vertical dashed lines at time $k = 200$ correspond to the end of the applied input sequence, from which the predictions and their uncertainty are illustrated by the orange lines and the shaded orange area.

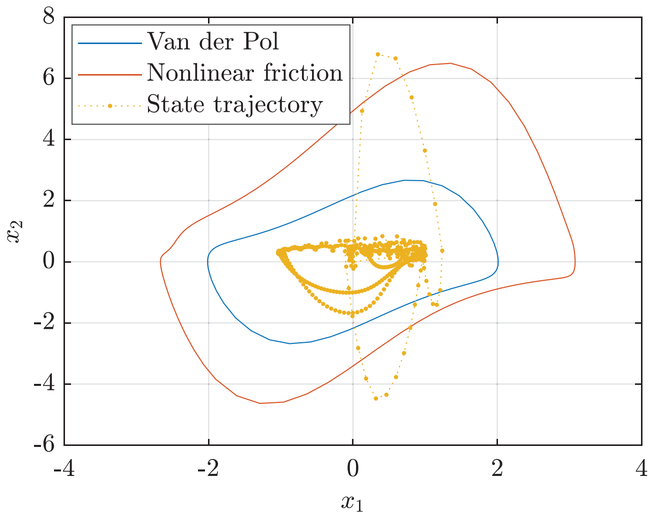


FIGURE 8 Phase diagram. The blue line illustrates the limit cycle of the Van der Pol system, whereas, the red cycle is obtained when the system is disturbed by the non-linear friction. The yellow curve illustrates the resulting state trajectory when the system is controlled by the proposed GP-MPC scheme.

AUTHOR CONTRIBUTIONS

Peter Polcz: Software, validation, writing - original draft. Tamas Péni: Conceptualization, methodology, supervision, writing - review and editing. Roland Tóth: Conceptualization, methodology, supervision, validation, writing - review and editing.

ACKNOWLEDGEMENTS

The research was supported by the European Union within the framework of the National Laboratory for Autonomous Systems (RRF-2.3.1-21-2022-00002) and by the Eötvös Loránd Research Network (grant. number: SA-77/2021).

CONFLICT OF INTEREST STATEMENT

The authors declare no conflict of interest.

DATA AVAILABILITY STATEMENT

The data that supports the findings of this study are available in the supplementary material of this article.

ORCID

Tamás Péni https://orcid.org/0000-0002-1440-4263

REFERENCES

1. Nguyen-Tuong, D., Peters, J.: Model learning for robot control: A survey. Cogn. Process. 12(4), 319–340 (2011). https://doi.org/10.1007/s10339-011-0404-1
2. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning), MIT Press, Cambridge, MA (2006)
3. Cucker, F., Smale, S.: On the mathematical foundations of learning. Bull. Am. Math. Soc. 39(1), 1–49 (2001). https://doi.org/10.1090/s0273-0979-01-00923-5
4. Klenske, E.D., Zeilinger, M.N., Schölkopf, B., Hennig, P.: Gaussian process-based predictive control for periodic error correction. IEEE Trans. Control Syst. Technol. 24(1), 110–121 (2016). https://doi.org/10.1109/TCST.2015.2420629
5. Grancharova, A., Kocijan, J., Johansen, T.A.: Explicit stochastic predictive control of combustion plants based on Gaussian process models. Automatica 44(6), 1621–1631 (2008). https://doi.org/10.1016/j.automatica.2008.04.002
6. Carron, A., Arcari, E., Wermelinger, M., Hewing, L., Hutter, M., Zeilinger, M.N.: Data-driven model predictive control for trajectory tracking with a robotic arm. IEEE Rob. Autom. Lett. 4(4), 3758–3765 (2019). https://doi.org/10.1109/LRA.2019.2929987
7. Nguyen-Tuong, D., Peters, J.: Incremental online sparsification for model learning in real-time robot control. Neurocomputing 74(11), 1859–1867 (2011). https://doi.org/10.1016/j.neucom.2010.06.033
8. Maiworm, M., Limon, D., Findeisen, R.: Online learning-based model predictive control with Gaussian process models and stability guarantees. Int. J. Robust Nonlinear Control 31(18), 8785–8812 (2021). https://doi.org/10.1002/rnc.5361

9. Hewing, L., Kabzan, J., Zeilinger, M.N.: Cautious model predictive control using Gaussian process regression. IEEE Trans. Control Syst. Technol. 28(6), 2736–2743 (2020). https://doi.org/10.1109/TCST.2019.2949757
10. Ostafew, C.J., Schoellig, A.P., Barfoot, T.D.: Conservative to confident: Treating uncertainty robustly within learning-based control. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, pp. 421–427. (2015). https://doi.org/10.1109/ICRA.2015.7139033
11. Candela, J.Q., Girard, A., Rasmussen, C.E.: Prediction at an uncertain input for Gaussian processes and relevance vector machines application to multiple-step ahead time-series forecasting. Tech. Report IMM-2003-18, Technical University of Denmark (2003)
12. Deisenroth, M.P., Huber, M.F., Hanebeck, U.D.: Analytic moment-based Gaussian process filtering. In: Proceedings of the 26th Annual International Conference on Machine Learning – ICML'09, pp. 225–232, ACM Press, New York (2009). https://doi.org/10.1145/1553374.1553403
13. Deisenroth, M.P.: Efficient reinforcement learning using Gaussian processes – Revised version, Ph.D. thesis. Faculty of Informatics Institute for Anthropomatics Intelligent Sensor-Actuator-Systems Laboratory (ISAS) (2017)
14. Nghiem, T.X.: Gaussian process derivative at uncertain input for SE kernel. Tech. report, School of Informatics, Computing, and Cyber Systems Northern Arizona University (2019). URL http://openknowledge.nau.edu/id/eprint/5501
15. Hewing, L., Liniger, A., Zeilinger, M.N.: Cautious NMPC with Gaussian process dynamics for autonomous miniature race cars. In: 2018 European Control Conference (ECC), pp. 1341–1348, IEEE, Piscataway, NJ (2018). https://doi.org/10.23919/ECC.2018.8550162
16. Blackmore, L., Açıkmeşe, B., Carson, J.M.: Lossless convexification of control constraints for a class of nonlinear optimal control problems. Syst. Control Lett. 61(8), 863–870 (2012). https://doi.org/10.1016/j.sysconle.2012.04.010
17. Mao, Y., Szmuk, M., Açıkmeşe, B.: Successive convexification of non-convex optimal control problems and its convergence properties. In: 2016 IEEE 55th Conference on Decision and Control (CDC), pp. 3636–3641, IEEE, Piscataway, NJ (2016). https://doi.org/10.1109/CDC.2016.7798816
18. Zanelli, A., Domahidi, A., Jerez, J., Morari, M.: Forces NLP: an efficient implementation of interior-point methods for multistage nonlinear nonconvex programs. Int. J. Control 93(1), 13–29 (2017). https://doi.org/10.1080/00207179.2017.1316017
19. Van Hessem, D., Bosgra, O.: Stochastic closed-loop model predictive control of continuous nonlinear chemical processes. J. Process Control 16(3), 225–241 (2006). https://doi.org/10.1016/j.jprocont.2005.06.003
20. Mesbah, A.: Stochastic model predictive control: An overview and perspectives for future research. IEEE Control Syst. Mag. 36(6), 30–44 (2016). https://doi.org/10.1109/MCS.2016.2602087
21. Nghiem, T.X.: Linearized Gaussian processes for fast data-driven model predictive control. In: 2019 American Control Conference (ACC), pp. 1629–1634, IEEE, Piscataway, NJ (2019). https://doi.org/10.23919/ACC.2019.8814476
22. Nghiem, T.X., Nguyen, T.-D., Le, V.-A.: Fast Gaussian process based model predictive control with uncertainty propagation. In: 2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton), pp. 1052–1059, IEEE, Piscataway, NJ (2019). https://doi.org/10.1109/ALLERTON.2019.8919857
23. González Cisneros, P.S., Werner, H.: Nonlinear model predictive control for models in quasi-linear parameter varying form. Int. J. Robust Nonlinear Control 30(10), 3945–3959 (2020). https://doi.org/10.1002/rnc.4973
24. Hespe, C., Werner, H.: Convergence properties of fast quasi-LPV model predictive control. In: 60th IEEE Conference on Decision and Control (CDC), pp. 3869–3874, IEEE, Piscataway, NJ (2021). https://doi.org/10.1109/CDC45484.2021.9683612
25. Hewing, L., Zeilinger, M.N.: Cautious model predictive control using Gaussian process regression, arXiv:1705.10702 (2017). URL http://arxiv.org/abs/1705.10702
26. Koelewijn, P.J.W., Tóth, R.: Automatic grid-based LPV embedding of nonlinear systems, Tech. report, Eindhoven University of Technology, (2021)
27. Tóth, R.: Modeling and Identification of Linear Parameter-Varying Systems, Springer, Berlin (2010). https://doi.org/10.1007/978-3-642-13812-6
28. Nguyen-Tuong, D., Seeger, M., Peters, J.: Computed torque control with nonparametric regression models. In: 2008 American Control Conference, pp. 212–217, IEEE, Piscataway, NJ (2008). https://doi.org/10.1109/ACC.2008.4586493
29. Andersson, J.A.E., Gillis, J., Horn, G., Rawlings, J.B., Diehl, M.: CasADi: A software framework for nonlinear optimization and optimal control. Math. Program. Comput. 11(1), 1–36 (2018). https://doi.org/10.1007/s12532-018-0139-4
30. Wächter, A., Biegler, L.T.: On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math. Program. 106(1), 25–57 (2005). https://doi.org/10.1007/s10107-004-0559-y
31. Amestoy, P., Duff, I.S., Koster, J., L'Excellent, J.-Y.: A fully asynchronous multifrontal solver using distributed dynamic scheduling. SIAM J. Matrix Anal. Appl. 23(1), 15–41 (2001)
32. Amestoy, P., Buttari, A., L'Excellent, J.-Y., Mary, T.: Performance and scalability of the block low-rank multifrontal factorization on multicore architectures. ACM Trans. Math. Software 45(1), 1–26 (2019)
33. MOSEK ApS: The MOSEK optimization toolbox for MATLAB manual. version 7.1 (revision 28) (2015). URL http://docs.mosek.com
34. Polcz, P.: Iterated LPV-MPC iteration of a GP-MPC problem using sequential quadratic programming, GitLab repository (2021). https://gitlab.com/ppolcz/Pr1_LPV_Impl_for_GP-MPC.git

How to cite this article: Polcz, P., Péni, T., Tóth, R.: Efficient implementation of Gaussian process–based predictive control by quadratic programming. IET Control Theory Appl. 17, 968–984 (2023). https://doi.org/10.1049/cth2.12430

APPENDIX

A.1 | Derivation of (20)

The derivation for (20) can be given in multiple steps, but it is essentially based on a simple observation, which allows expressing the expectation of the squared weighted norm of a random variable x as

$$\mathbb{E}\left(\|x\|_Q^2\right) = \|\mu_x\|_Q^2 + \mathrm{Tr}(Q\Sigma_x). \tag{A.1}$$

To prove (A.1), first, consider the following chain of identities:

$$Q\Sigma_x = Q\mathrm{Var}(x) = \mathrm{Cov}(Qx, x) = \mathbb{E}(Qxx^\top) - Q\mu_x(\mu_x)^\top.$$

Then, we take the trace of the expression above to obtain:

$$\mathrm{Tr}(Q\,\Sigma_{\mathrm{x}}) = \mathbb{E}(x^\top Q\,x) - (\mu_{\mathrm{x}})^\top Q\,\mu_{\mathrm{x}}. \tag{A.2}$$

Equality (A.1) is a direct consequence of (A.2). Accordingly, the squared weighted norm of the output error at time $k+i$ predicted at time k can be expressed as follows:

$$\|Cx(i|k) - r(k+i)\|_Q^2 = \|C\mu_{\mathrm{x}}(i|k) - r(k+i)\|_Q^2 + \mathrm{Tr}\big(Q\,C\,\Sigma_{\mathrm{x}}(i|k)\,C^\top\big). \tag{A.3}$$

Secondly, the input cost is expressed as follows:

$$\mathbb{E}\big(\|u(i|k)\|_R^2\big) = \mathbb{E}\big(\|\mu_{\mathrm{u}}(i|k) - K\left(x(i|k) - \mu_{\mathrm{x}}(i|k)\right)\|_R^2\big) = \|\mu_{\mathrm{u}}(i|k)\|_R^2 + \mathrm{Tr}(R\,\mathcal{K}\,\Sigma_{\mathrm{x}}(i|k)\,\mathcal{K}^\top). \tag{A.4}$$

The same expression for the expectation have been reported in the literature, for example, in [25] although without the provided explanation given above.