Learning-based SMPC for Reference Tracking under State-dependent Uncertainty: An Application to Atmospheric Pressure Plasma Jets for Plasma Medicine

Angelo D. Bonzanini, David B. Graves, and Ali Mesbah

***Abstract*— The increasing complexity of modern technical systems can exacerbate model uncertainty in model-based control, posing a great challenge to safe and effective system operation under closed-loop. Online learning of model uncertainty can enhance control performance by reducing plant-model mismatch. This paper presents a learning-based stochastic model predictive control (LB-SMPC) strategy for reference tracking of stochastic linear systems with additive state-dependent uncertainty. The LB-SMPC strategy adapts the state-dependent uncertainty model online to reduce plant-model mismatch for control performance optimization. Standard reachability and statistical tools are leveraged along with the state-dependent uncertainty model to develop a chance constraint-tightening approach, which ensures state constraint satisfaction in probability. The stability and recursive feasibility of the LB-SMPC strategy are established for tracking time-varying targets, without the need to redesign the controller every time the target is changed. The performance of the LB-SMPC strategy is experimentally demonstrated on an atmospheric pressure plasma jet (APPJ) testbed with prototypical applications in plasma medicine and materials processing. Real-time control comparisons with learning-based MPC with no uncertainty handling and offset-free MPC showcase the usefulness of LB-SMPC for predictive control of safety-critical systems with hard-to-model and/or time-varying dynamics.**

I. Introduction

Atmospheric pressure plasma jets (APPJs) are a class of non-equilibrium plasma devices used for treatment of complex surfaces in applications such as processing of (bio)materials [1] and plasma medicine, for example, to combat antibiotic-resistant bacteria [2], shrink cancerous tumors [3], and accelerate healing of chronic wounds [4]. In APPJs, the plasma is typically generated by applying an electric field to a noble gas (usually Argon or Helium) that flows in a dielectric tube [5]. The observed therapeutic effects of the plasma are postulated to result from the synergy between a multitude of effects such as short-lived reactive chemical species, UV photons, and electric and thermal effects [6], [7].

Despite their promise, harnessing the medical potential of plasmas in a systematic and reproducible manner remains a challenge. APPJs are typically hand-held devices. Therefore, their operation relies on user expertise to judge treatment effectiveness, while avoiding undesirable consequences. This inexact mode of operation is exacerbated by the intrinsic variability and nonlinear behavior of the plasma. Specifically, APPJs suffer from run-to-run variations, even when the operating conditions are nearly identical [8], while also exhibiting sharp radial and axial gradients in both temperature and concentration of reactive species [9]. Thus, APPJs are overly sensitive to exogenous disturbances, such as tip-to-surface separation distance or variations in the properties of the target surface.

Recently, Gidon et al. demonstrated the effectiveness of model predictive control (MPC) strategies for controlling the delivery of plasma effects to a surface [10], [11]. However, these works did not account for system uncertainty, which is particularly important given the safety-critical nature of APPJ applications in plasma medicine. Although nominal MPC can provide some degree of robustness due to its receding-horizon implementation, it cannot guarantee robust feasibility (i.e., robust constraint satisfaction and recursive feasibility) of the controller in the presence of uncertainties [12]. In addition, the reported MPC strategies for APPJs are based on relatively simple physics-based or data-driven models. Such models are generally incapable of capturing (i) the complex and hard-to-model dynamics of plasma and plasma-surface interactions and (ii) the time-varying nature of surface properties, such as the electrical, thermal, and chemical properties of biological tissues during a plasma treatment [13], [14], [15], [16].

This paper aims to investigate learning-based MPC (LB-MPC) of APPJs. Learning-based control [17], [18] holds promise for reducing the discrepancy between "simple" control-relevant models and the actual complex and not well-understood dynamics of plasmas in real-time [14]. The promise of LB-MPC has been demonstrated in various applications, such as pH neutralization processes [19], gas-liquid separation plants [20], and robot path tracking [21]. The notion of LB-MPC with robustness guarantees was first introduced in [22]. The key idea is that safety and performance are decoupled by using two models of the system: an approximate model with bounds on uncertainty and a second model that is updated online using statistical methods. The rationale behind this decoupling is to use the first model to establish safety and robustness guarantees offline using reachability tools, and the second model for improving control performance online. This idea was further investigated in [23] using a non-parametric machine learning technique, where LB-MPC approaches with guaranteed sta-

This work was supported by the National Science Foundation under Grant 1839527.

A. D. Bonzanini, David B. Graves, and Ali Mesbah are with the Department of Chemical and Biomolecular Engineering, University of California, Berkeley, CA 94720, USA (email: {adbonzanini, graves, mesbah}@berkeley.edu).

bility by design were proposed. Recently, Gaussian process (GP) regression [24] has proven to be particularly well-suited for LB-MPC since its non-parametric form lends itself well for modeling the mismatch between the nominal system model and the real dynamics, which is generally hard to parametrize. This can be done, for example, by learning a state- and input-dependent uncertainty term from the data [25]. In particular, the case of LB-MPC using GP with robustness guarantees was recently investigated in [26], [27], where GP regression is used to correct for model uncertainty. In [26], the GP model is used exclusively offline to derive tightened constraints, which provide robustness guarantees of the MPC. In [27], additional approximations are used for efficient computation of the GP online, which is used both for tightening the chance constraints online (using the state mean and covariance propagation equations) and for performance optimization (by correcting the model predictions). LB-MPC using GP regression has also been investigated in the context of scenario-tree MPC, whereby the GP model is used to adapt the scenarios online, therefore reducing the conservativeness associated with standard scenario-based MPC approaches [28]. For a recent review on LB-MPC with a focus on safe learning, the reader is referred to [25].

Motivated by challenges in predictive control of APPJs for treatment of complex surfaces with time-varying properties, this paper presents a learning-based stochastic MPC (LB-SMPC) strategy for reference tracking of stochastic linear systems with state-dependent uncertainty. Leveraging the framework established in [22], we use two models of the system, i.e., an approximate model with bounds on the uncertainty and a second model that is updated online, to extend the provably safe and robust LB-MPC strategy [22] to chance-constrained stochastic linear systems with additive state-dependent uncertainty. Specifically, the theoretical contributions of this work include: (i) we combine standard reachability tools from tube-based MPC with the learned state-dependent uncertainty model to extend the constraint tightening method presented in [26] to guarantee chance constraint satisfaction under state-dependent uncertainty; (ii) we extend the LB-SMPC approach to enable reference tracking of reachable targets without the need to redesign the controller every time the target is changed, which also enlarges the region of attraction (ROA) of the controller [29]; and (iii) we prove stability and recursive feasibility for the LB-SMPC strategy for reference tracking.

In addition, we experimentally test the proposed LB-SMPC strategy on a medically-motivated APPJ case study in which we leverage the GP model to effectively control the plasma treatment properties when surface characteristics are changed. Such cases arise, for example, due to the intrinsic variabilities of biological tissues, or the surfaces changes induced as a result of the plasma treatment. We experimentally demonstrate the improvement in closed-loop performance due to the learning of plant-model mismatch, while guaranteeing state constraint satisfaction that is critical for safety-critical applications. Furthermore, online learning of the state-dependent uncertainty is explored to mimic a situation in which there is no historical data on the patient undergoing plasma treatment.

The paper is organized as follows. Section II introduces the APPJ testbed and defines the problem setup. Section III consolidates existing contributions in the tube-based MPC literature to develop the proposed state-dependent chance-constraint tightening approach for guaranteeing constraint satisfaction up to a pre-defined probability level. Section IV formulates the LB-SMPC problem for reference tracking and presents the stability and recursive feasibility results. Section V discusses the results of closed-loop simulations and real-time control experiments. Finally, section VI concludes the paper.

Notation. The set of real numbers is denoted by $\mathbb{R}$, while the set of (positive) integers as $(\mathbb{Z}_+)$ $\mathbb{Z}$. The quadratic norm of x with respect to $Q = Q^\top \succ 0$ is defined as $||x||_Q^2 = x^\top Q x$. A vector-valued variable $x \sim \mathcal{N}(\mu, \Sigma)$ represents a normally distributed variable with mean $\mathbb{E}[x] = \mu$, where $\mathbb{E}[\cdot]$ denotes the expectation operator, and covariance Σ. Nominal state predictions are denoted by $\tilde{x}$, whereas measured states are denoted by x. The probability of an event A occurring is denoted as $\mathbb{P}[A]$. When a constraint set $\mathbb{X}$ is tightened, it is written as $\overline{\mathbb{X}}$.

II. Problem Statement

In this section, we present the APPJ testbed and a control-oriented model for plasma treatment of surfaces. We also describe the formulation of the control problem.

A. APPJ Testbed

The kHz-excited APPJ in Helium (He) consists of a copper ring electrode wrapped around a dielectric quartz tube, as shown in Fig. 1. To generate the plasma, a sinusoidal high-voltage electric field is applied to the copper ring electrode. He flows through the tube and is directed towards a grounded, glass-covered metal plate, 4 mm below the tip of the tube. The surface can be changed by removing the glass cover, thereby exposing the metal surface underneath

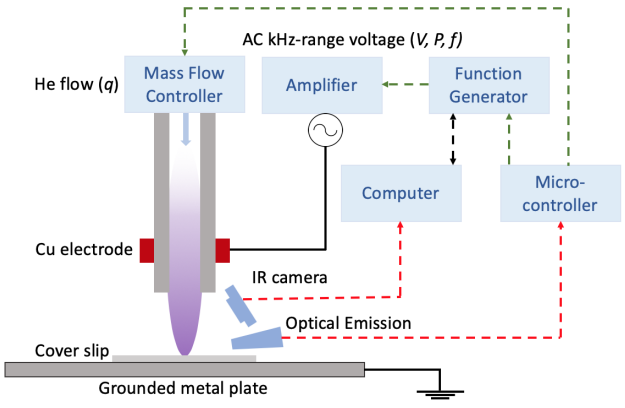

Fig. 1: Schematic of the kHz-excited APPJ in He. Red dotted lines indicate measured outputs (surface temperature and optical plasma intensity). Green dotted lines indicate the manipulated inputs (He flow rate and applied power).

it. Changing the target surface markedly affects the plasma-surface interactions due to the vast difference in conductivity between glass and metal.

The manipulated inputs of the APPJ include the flow rate of He q (slm) and the applied power P (W). A microcontroller (Arduino UNO) coupled with a controllable function generator (XR-2602CP) is used to generate a sinusoidal waveform at a given frequency. This signal is amplified through an amplifier (TREK). The flow rate is manipulated through a mass flow controller. Measured outputs of the APPJ include the maximum surface temperature T (°C) and optical plasma intensity incident on the surface I (a.u.). Surface temperature is measured through a thermal camera (FIR Lepton), while the optical plasma intensity is measured through a fiber optic cable connected to an optical emission spectrometer (OES). Thermal images are processed on a single-board computer (Raspberry Pi 3) to determine the maximum surface temperature. The entire data acquisition process is coordinated in Python. Measurements are taken at relatively fast timescales (≈ 1 s). Sensor failures occasionally occur due to communication or synchronization errors between the hardware and the software. Such measurements are removed and replaced with an average of the previous and next measurements.

B. Control-relevant APPJ Modeling

APPJs are notoriously hard to model, since they exhibit nonlinear dynamics that are distributed across multiple length- and time-scales. This modeling challenge is further exacerbated by the plasma's intrinsic variability and the APPJ's sensitivity to exogenous disturbances, such as tip-to-surface separation distance. Moreover, real-time solution of first-principles models of plasmas, which generally consist of a set of multi-dimensional partial differential equations [30], is not practical for online control of the plasma effects on complex surfaces that typically occur on milliseconds to seconds timescales.

Here, we resort to data-driven modeling of the APPJ shown in Fig. 1. Multi-step tests were performed on a glass surface around the desired nominal operating condition, which is defined to be $T^s = 38.5$ °C, $I^s = 100$ a.u., $P^s = 3.0$ W, and $q^s = 3.0$ slm; superscript s denotes the nominal (steady-state) condition [5]. The input-output data were used for subspace identification of a linear multi-input multi-output model [31]. The state-space model takes the form

$$x_{k+1} = Ax_k + Bu_k \tag{1a}$$
$$y_k = Cx_k + Du_k, \tag{1b}$$

with states $x \in \mathbb{R}^{n_x}$, inputs $u \in \mathbb{R}^{n_u}$, controlled variables $y \in \mathbb{R}^{n_y}$, and time-step $k \in \mathbb{N}$. The state-space model is defined in terms of deviation variables around the nominal operating point, i.e., $y = [T - T^{\text{s}},\ 0.1(I - I^{\text{ss}})]^\top$ and $u = [q - q^{\text{s}},\ P - P^s]^\top$. We consider an observable canonical form of (1) with $C = I$ and $D = \mathbf{0}$. The subspace identification was performed in MATLAB using the `n4sid` function. The state-space matrices are reported in Appendix A.

To account for the mismatch between the linear state-space model (1) and the nonlinear and time-varying plasma dynamics that mainly arise from variations in electrical properties of the surface in the considered case study, we describe the APPJ dynamics by

$$x_{k+1} = Ax_k + Bu_k + B_d\left(g(x_k) + w_k\right), \tag{2}$$

where $B_d \in \mathbb{R}^{n_x \times n_w}$, $w_k \in \mathbb{R}^{n_w}$ is a process noise that is assumed to be normally distributed, i.e., $w_k \sim \mathcal{N}(0, \Sigma^w)$, and $g(x) : \mathbb{R}^{n_x} \to \mathbb{R}^{n_w}$ is a state-dependent, vector-valued function that describes the unmodeled system dynamics. $g(x)$ is to be learned from data. For ease of notation, we define the overall system uncertainty as a stochastic variable $\tilde{v}(x) := B_d(g(x) + w)$.

Remark 1: System description (2) provides an advantageous model structure because it offers the flexibility of learning the general (possibly nonlinear) uncertainty description $g(x)$ from data while it incorporates a linear state-space model as a nominal system model. This structure enables decoupling safety and robustness aspects of the controller design from performance [22]. The simple linear structure of the nominal model allows using well-known reachability tools to provide guarantees for safety and robustness, while the additive uncertainty term $g(x)$ is learned from data to update the model for performance optimization.

We use the previously collected $M > 0$ data points in subspace identification to learn a model for $g(x)$ using GP regression [24]. This enables quantifying the uncertainty of model predictions in the form a covariance matrix, from which confidence intervals for model uncertainty can be derived. To generate the training dataset, we evaluate the mismatch term as

$$y_i^{\text{mis}} = B_d^\dagger\left(x_{i+1} - (Ax_i + Bu_i)\right) = g(x_i) + w_i, \tag{3}$$

$\forall i = 1, \ldots, M$, where $B_d^\dagger$ is the Moore-Penrose pseudoinverse of B_d. Then, the training dataset becomes

$$\mathcal{D} = \left\{\mathbf{y} = \left[y_1^{\text{mis}}, \ldots, y_M^{\text{mis}}\right]^\top, \mathbf{x} = \left[x_0, \ldots, x_{M-1}\right]^\top\right\}.$$

For notational convenience, it is assumed that each dimension of $g(x)$ is learned separately.

Furthermore, we introduce the stochastic variable $d(x)$, which aims to model $g(x)$. To construct the GP model $d(x)$, we first specify a prior on each element $a = 1, \ldots, n_w$ of $g(x)$ with a known mean function $m^a(x)$, which is typically assumed to be zero in the absence of any other information, and covariance kernel $k^a(x_i, x_j)$. The Gaussian posterior distribution can be derived by conditioning on the dataset $\mathcal{D}$ at any specified test point x. Since the posterior distribution is Gaussian, it suffices to specify the conditional mean and covariance as

$$\mu_a^d(x) = m^a(x) + K_{x\mathbf{x}}^a\left(K_{\mathbf{xx}}^a + \sigma_a^2 I\right)^{-1}\left([\mathbf{y}]_a - m_{\mathbf{x}}^a\right) \tag{4a}$$
$$\Sigma_a^d(x) = K_{xx}^a - K_{\mathbf{xx}}^a\left(K_{\mathbf{xx}}^a + \sigma_a^2 I\right)^{-1} K_{\mathbf{x}x}^a, \tag{4b}$$

where K is the Gram matrix that is composed of the terms $[K^a_{\mathbf{xx}}]_{i,j} = k^a(x_i, x_j)$, $[K^a_{\mathbf{x}x}]_i = k^a(x_i, x)$, $K^a_{x\mathbf{x}} = (K^a_{\mathbf{x}x})^\top$, $[K^a_{xx}] = k^a(x, x)$, $[m^a_{\mathbf{x}}]_i = m^a(x_i)$, and σ_a is the a^{th} component of the diagonal of Σ^w, which is known.

Remark 2: Non-Markov processes can also be modeled by keeping track of a larger number of GP inputs and outputs. Accordingly, the training dataset becomes $\mathcal{D} = \{[\mathbf{y}_{k-\ell}, \ldots, \mathbf{y}_{k-1}], [\mathbf{x}_{k-\ell}, \ldots, \mathbf{x}_{k-1}]\}$, where ℓ is the lag, i.e., how many previous time steps we are using to make the prediction at time k.

A popular choice for the kernel function is the squared exponential kernel

$$k^a(x_i, x_j) = \sigma^2_{f,a} \text{exp}\left(-\frac{1}{2}(x_i - x_j)^\top L_a^{-1}(x_i - x_j)\right),$$

where $\sigma^2_{f,a}$ is the signal variance and L_a is a positive diagonal length-scale matrix [24]. The resulting state-dependent model of the unknown function $g(x)$ is given by

$$d(x) \sim \mathcal{N}\left(\mu^d(x), \Sigma^d(x)\right), \tag{5}$$

where the individual predictions in (4) are concatenated as $\mu^d(x) = \left[\mu^d_1(x), \ldots, \mu^d_{n_w}(x)\right]^\top$ and $\Sigma^d(x) = \text{diag}\left(\left[\Sigma^d_1(x), \ldots, \Sigma^d_{n_w}(x)\right]^\top\right)$. We define the overall predicted source of uncertainty as

$$\tilde{w}(x) := B_d\left(d(x) + w\right) \sim \mathcal{N}\left(\mu^{\tilde{w}}(x), \Sigma^{\tilde{w}}(x)\right), \tag{6}$$

where $\mu^{\tilde{w}}(x) = B_d\left(\mu^d(x) + 0\right)$ and $\Sigma^{\tilde{w}}(x) = B_d\left(\Sigma^d(x) + \Sigma^w\right) B_d^\top$.

We assume that the distribution of $\tilde{w}(x)$ has a bounded support, i.e., $\tilde{w}(x) \in \widetilde{\mathbb{W}}(x)$, where $\widetilde{\mathbb{W}}(x)$ is a compact state-dependent set. However, due to the GP approximation of $g(x)$ in combination with the Gaussian distributed process noise w, the bounded support assumption of $\widetilde{\mathbb{W}}(x)$ is not automatically satisfied. We use confidence intervals to define a practical bounded range for the overall uncertainty, which can be computed as

$$\widetilde{\mathbb{W}}(x) := \left\{\tilde{w} : ||\tilde{w} - \mu^{\tilde{w}(x_k)}||^2_{\Sigma^{\tilde{w}}(x)} \leq r(\vartheta)\right\}, \tag{7}$$

where $r(\cdot)$ is the quantile function of a chi-squared distribution with n_w degrees of freedom, $\chi^2_{n_d}(\vartheta)$, and $\vartheta \in (0, 1)$ is the desired probability level. In this work, we choose a practical bound $\vartheta = 0.99$. The prediction $d(x)$ of the unmodeled dynamics $g(x)$ is validated against a previously unseen dataset, as shown in Appendix A.

Remark 3: The proposed method for modeling the plant-model mismatch assumes that successive GP function evaluations are independent. Therefore, it does not take into account the potential correlation of successive GP function evaluations, as discussed in [32] and [33]. This interpretation corresponds to considering the overall uncertainty $\tilde{v}(x)$ as process noise [33]. Considering successive GP function evaluations to be independent reduces the computational complexity and enables evaluating the GP model in prediction.

C. Control Problem for the APPJ

We aim to design a LB-SMPC controller for the APPJ described by (2), where the state-dependent system uncertainty is modeled by (6). The control objective is to track a reference trajectory in the surface temperature T and plasma optical intensity I in the presence of significant plant-model mismatch that arises from the time-varying nature of the target surface. The time-varying surface properties mimic treatment of patients with different tissue properties, or even treatment of the same patient but on different/changing tissues due to plasma treatment. Hence, the LB-SMPC is intended to mitigate the plant-model mismatch by adapting the model of the state-dependent uncertainty $g(x)$ using real-time data.

The LB-SMPC strategy must ensure adherence to input and state constraint satisfaction under model uncertainty. For example, in plasma medicine, an important constraint is that of surface temperature, since it is closely related to thermal dose accumulation on the surface [11]. Exceeding pre-specified surface temperature constraints can compromise patient safety by irreversibly damaging cells, while also deteriorating patient comfort during the treatment. We define state and input constraints as polytopic sets, denoted by $\mathbb{X}$ and $\mathbb{U}$, respectively, i.e.,

$$x \in \mathbb{X}, \tag{8a}$$

$$u \in \mathbb{U}. \tag{8b}$$

For compactness of notation, we define $\mathbb{Z} = \mathbb{X} \times \mathbb{U}$. In addition, we impose individual chance constraints (ICCs) on states as

$$\mathbb{P}[h_i^\top x_{k+1} \leq 1] \geq 1 - \varepsilon_i, \; i = 1 \ldots, n_c, \tag{9}$$

where ε_i is the probability of constraint violation of the i^{th} chance constraint, h_i is a vector of constants, and n_c is the total number of ICCs. Notice that, since the state-space model (1) is identified in the observable canonical form where $C = I$ and $D = \mathbf{0}$, it follows that $y_k = x_k$ and therefore state constraints are equivalent to output constraints.

In the next section, we introduce the relevant theory needed to develop the state-dependent constraint tightening approach for the proposed LB-SMPC strategy.

III. Constraint Tightening for Linear Systems with Stochastic State-dependent Uncertainty

This section first summarizes existing results on characterization of steady states for reference tracking and dual mode prediction for constraint handling. This is followed by an extension of the robust constraint tightening method presented in [26] to allow for handling of chance constraints for linear systems with state-dependent uncertainty. The following assumptions are made for the remainder of the paper.

Assumption 1:

i) The pair (A,B) is controllable.

ii) Sets $\mathbb{X}$, $\mathbb{U}$, and $\widetilde{\mathcal{W}} = \cup_{x\in\mathbb{X}}\widetilde{\mathbb{W}}(x)$ contain the origin in their interior.
iii) Sets $\mathbb{U}$ and $\widetilde{\mathcal{W}}$ are compact and convex.
iv) The system state x_k is measured.

A. Characterization of Steady States

One of the control objectives is to track reference trajectories that may change over time. At steady state, there is no dynamic evolution of the nominal system

$$\tilde{x}_{k+1} = A\tilde{x}_k + Bu_k,$$

i.e., $\tilde{x}_{k+1} = \tilde{x}_k = \tilde{x}$. Therefore, every pair of steady states and inputs $\tilde{z}^s = [\tilde{x}^s, \tilde{u}^s]^\top$ must be a solution of the following set of linear equations

$$\begin{bmatrix} A - I_n & B \end{bmatrix} \begin{bmatrix} \tilde{x}^s \\ \tilde{u}^s \end{bmatrix} = 0,$$

and thus an element of the null space of matrix $\begin{bmatrix} A - I_n & B \end{bmatrix}$ [34]. Since (A, B) is controllable by Assumption 1, the dimension of this null space is m. Consequently, there exists a matrix $M_\theta \in \mathbb{R}^{(n_x+n_u)\times n_y}$ such that

$$\tilde{z}^s = M_\theta \tilde{\theta}, \tag{10a}$$
$$\tilde{y}^s = N_\theta \tilde{\theta}, \tag{10b}$$

for any $\theta \in \mathbb{R}^{n_y}$ and $N_\theta = [C\ D]\, M_\theta$. Since we have identified the APPJ model such that $y_k = x_k$, N_θ serves the purpose of selecting the appropriate elements that correspond to the steady-states of x, i.e., $\tilde{y}^s = \tilde{x}^s = N_\theta\tilde{\theta}$. The goal of parametrization (10) is to enlarge the terminal invariant set for tracking when compared to regulation to a fixed target [34]. As a result, the controller will exhibit a larger region of attraction (ROA), ensuring feasibility when targets change [29]. A detailed method for computing matrices M_θ and N_θ is presented in [35].

B. Dual Mode Prediction

The dual mode prediction paradigm ensures that state constraints are satisfied for an infinite prediction horizon, while convergence properties of the controller are guaranteed. Predictions of system (2) at time k are represented by

$$x_{j+1|k} = Ax_{j|k} + Bu_{j|k} + \tilde{w}(x_{j|k}), \tag{11}$$

where $\tilde{w}(x_{j|k})$ denotes the predicted state-dependent uncertainty as defined in (6).

We aim to define a terminal feedback controller such that it drives the system as close as possible to a desired target $u_{j|k} = \tilde{u}_k^s + K\left(x_{j|k} - \tilde{x}_k^s\right)$, where K can be chosen as the infinite-horizon linear quadratic regulator (LQR) gain. Using the steady-state characterization (10), a control law parametrization can be devised as[1]

$$u_{j|k} = \pi(x_{j|k}, c_{j|k}) = Kx_{j|k} + L\tilde{\theta}_k + c_{j|k}, \tag{12}$$

where $c_{j|k}$ are the $n_u N$ decision variables, N is the prediction horizon, and $L = [-K\ I_m]\, M_\theta$ is a known matrix [29]. Substituting the parametrized control law (12) into (11) yields the prediction of the closed-loop dynamics

$$x_{j+1|k} = A_K x_{j|k} + B\left(L\tilde{\theta}_k + c_{j|k}\right) + \tilde{w}(x_{j|k}), \tag{13}$$

where $A_K = A + BK$, with eigenvalues strictly inside the unit circle. Furthermore, we define the nominal closed-loop system predictions by neglecting the state-dependent uncertainty in (13)

$$\tilde{x}_{j|k} = A_K\tilde{x}_{j|k} + B\left(L\tilde{\theta}_k + c_{j|k}\right). \tag{14}$$

Accordingly, we modify the constraint set $\mathbb{Z}$ in (8) to

$$\begin{bmatrix} x_k^\top & c_k^\top \end{bmatrix}^\top \in \mathbb{Z}_\pi. \tag{15}$$

The idea behind dual mode prediction is to switch between a receding-horizon controller and a local linear controller depending on whether the state is inside or outside the terminal region [36]. In mode 1, which includes steps $k \in [0, N-1]$, the decision variables $c_{j|k}$ are free to vary. In mode 2, which includes $k \in [N, \infty)$, $c_{j|k}$ is set to 0, effectively switching to a linear controller of the form $u_{j|k} = u_k^s + K(x_{j|k} - x_k^s)$. In mode 2, constraints are enforced by ensuring that the system is within the terminal region $\mathbb{X}_f$ [37]. By definition, the terminal set is constructed such that when the system converges to $x_N \in \mathbb{X}_f$, the closed-loop dynamics under the terminal control law are guaranteed to stay within $\mathbb{X}_f$, i.e.,

$$x_k \in \mathbb{X}_f \Rightarrow x_{k+1} = A_K x_k + BL\tilde{\theta}_k + \tilde{w}(x_k) \in \mathbb{X}_f.$$

C. Chance Constraint Tightening Under State-dependent Uncertainty

The idea behind recursive constraint tightening is to ensure that if the nominal system (14) lies within tighter state and input constraint sets $\overline{\mathbb{X}}_j$ and $\overline{\mathbb{U}}_j$, then the closed-loop system is guaranteed to satisfy the original constraints $\mathbb{X}$ and $\mathbb{U}$ in (8) [38]. Here, we look to modify the state-dependent robust constraint tightening approach presented in [26] to ensure chance constraint satisfaction (9). The main difference between the proposed chance constraint tightening method and the robust constraint tightening method presented in [26] lies in the construction of the first tightened constraint set $\overline{\mathbb{X}}_1$. In principle, ICCs (9) can be reformulated into deterministic constraints by defining a new set $\mathbb{X}^*$ as [39]

$$\mathbb{X}^* = \left\{x \in \mathbb{R}^n : h_i^\top x \le 1 - \gamma_i^*(x)\right\}, \tag{16}$$
$$\text{where} \quad \gamma_i^*(x) = \min \gamma_i \quad \text{s.t. } \mathbb{P}[h_i^\top \tilde{v}(x) \le \gamma_i] \ge 1 - \varepsilon_i,$$

where $i = 1, \ldots, n_c$, $\tilde{v}(x) = B_d(g(x) + w)$, and $\gamma_i^*(x)$ are state-dependent backoff parameters. The goal is to use these backoff parameters in combination with the state-dependent uncertainty $\tilde{v}(x)$ to recursively tighten the constraint set $\mathbb{X}^*$.

We denote j-step ahead tightened constraint sets by $\overline{\mathbb{X}}_j$. To generate the tightened constraint sets $\overline{\mathbb{X}}_j$ offline, we first compute the initial set $\overline{\mathbb{X}}_1 = \mathbb{X}^*$ by determining the backoff parameters $\gamma_i^*(x)$. Since the probability distribution of $\tilde{v}(x)$ is not known, we use the empirical cumulative distribution function (ECDF) of $\tilde{v}(x)$ (through its samples)

[1]A linear feedback parametrization is typically chosen because of its simplicity and computational tractability, although, other parametrizations are equally valid.

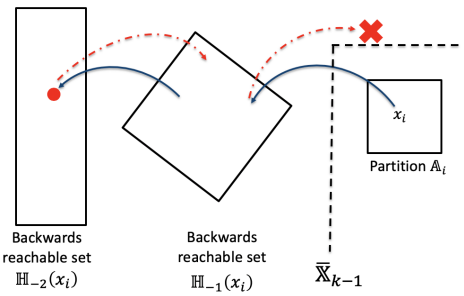

Fig. 2: Schematic of backwards reachable set computation.

to construct the initial tightened constraint set $\mathbb{X}^*$ [40]. From (16), it is apparent that any $\gamma_i(x)$, say, $\hat{\gamma}_i(x)$, will satisfy $\hat{\gamma}_i(x) \geq \gamma_i^*(x)$, since $\gamma_i(x)$ is a minimum. An estimate of $\hat{\gamma}_i$ can be obtained from the ECDF of $\tilde{v}(x)$. Consider $N_s = N_s^x \times N_s^v$ normalized uncertainty samples $h_i^\top \tilde{V} := \left\{h_i^\top \tilde{\mathbf{v}}(x_1), h_i^\top \tilde{\mathbf{v}}(x_2), \ldots, h_i^\top \tilde{\mathbf{v}}(x_{N_s^x})\right\}$, where $\tilde{\mathbf{v}}(x_j)$ denotes a collection of samples of $\tilde{v}(x_j)$, i.e., $\tilde{\mathbf{v}}(x_j) := \left\{\tilde{v}_1(x_j), \tilde{v}_2(x_j), \ldots, \tilde{v}_{N_s^v}(x_j)\right\}$. That is, we sample the state-space into N_s different samples and obtain an empirical probability distribution at each state x_i. The ECDF of the stochastic process $h_i^\top \tilde{V}$ can be defined as

$$\hat{F}_{h_i^\top \tilde{V}}(q, x) = \frac{1}{N_s^v} \sum_{p=1}^{N_s^v} \mathbb{1}\left(h_i^\top \tilde{v}_p(x) \leq q\right),$$

$$\text{where } \mathbb{1}\left(h_i^\top \tilde{v}_p(x) \leq q\right) = \begin{cases} 1, & \text{if } h_i^\top \tilde{v}_p(x) \leq q, \\ 0, & \text{otherwise.} \end{cases}$$

Thus, the backoff $\gamma_i^*(x)$ can be estimated from

$$\gamma_i^*(x) \leq \hat{\gamma}_i(x) = \hat{F}^{-1}_{h_i^\top \tilde{V}}(1-\varepsilon, x). \tag{17}$$

Then, the constraint tightening can be implemented as described in [26], whereby the state-space is partitioned into subsets, which are either conserved or discarded according to the uncertainty propagation of the samples x_i within them. This allows for computing the initial tightened state constraint set as

$$\mathbb{X}^* = \overline{\mathbb{X}}_1 = \left\{x \in \mathbb{R}^n : h_i^\top x \leq 1 - \hat{\gamma}_i(x)\right\}, \tag{18}$$

for $i = 1, \ldots, n_c$. Note that the initial constraint tightening step based on the ECDF of $\tilde{v}(x)$ can be used even when the random variables are correlated [41]. Furthermore, the model of $\tilde{v}(x)$, i.e., $\tilde{w}(x)$, need not be used at all in this first step, as long as there is data available for $\tilde{v}(x)$.

After constructing $\overline{\mathbb{X}}_1$ according to (18), existing robust constraint tightening methods can be used to tighten the subsequent constraint sets. In this work, we make use of the state-dependent constraint tightening method presented in [26]. First, we provide a brief outline of this method. The key idea is to compute backwards reachable sets $\mathbb{H}_{-j}(x)$, for all states $x \in \mathbb{X}$, for N time instances. Then, we can estimate the maximum admissible uncertainty by considering the worst-case uncertainty propagation from the backward reachable sets using the state-dependent uncertainty model (6), i.e., $\max_{\forall x \in \mathbb{H}_{-j}(x)}\{\tilde{w}(x)\}$. If the state-dependent worst-case system evolution results in a state outside of the previous tightened constraint set $\overline{\mathbb{X}}_{j-1}$, x_i is discarded, further tightening the constraint set $\overline{\mathbb{X}}_j$. The main difference in the chance-constrained setting is that we start from $\mathbb{X}^* = \overline{\mathbb{X}}_1$, rather than $\mathbb{X}$.

Since computing backwards reachable sets for all the individual states is prohibitively expensive, the state-space can be partitioned into subsets $\mathbb{A}_m$, $m = 1, \ldots, n_A$. A useful practical approximation is to describe $\overline{\mathbb{X}}_1$ with a union of polytopes. This will remove any curvature along the edges and will make the set computations easier since one can work exclusively with polytopes, for which efficient toolboxes exist, e.g., [42]. After partitioning the state-space into subsets $\mathbb{A}_m$, the computation of backwards reachable sets can be performed for all subsets $\mathbb{A}_m$ (or a discrete sample of states within subset $\mathbb{A}_m$), instead of all states x. Then, it is straightforward to maximize the uncertainty within $\mathbb{H}_{-j}(\mathbb{A}_m)$ since we have an expression for $\tilde{w}(x)$ from the GP model (6). This is summarized in Algorithm 1 in [26], and a conceptual schematic is shown in Fig. 2. Note that $\max_{x \in \mathbb{H}_{-j}(\mathbb{A}_m)}\{\tilde{w}(x)\} \leq \max_{x \in \mathbb{X}}\{\tilde{w}(x)\}$, which leads to less conservative constraint tightening than assuming a constant (worst-case) uncertainty over the entire state-space.

Finally, to tighten the hard input constraints (8b), a similar procedure can be followed. Given that $u_{j|k} = \tilde{u}_k^s + K(x_{j|k} - \tilde{x}_k^s)$, we can generate the tightened input constraints by determining whether the parametrized control input, which corresponds to the state-dependent worst-case state evolution, lies outside the previous tightened constraint set $\overline{\mathbb{U}}_{j-1}$.

Remark 4: There is a trade-off between reducing conservativeness in the state-dependent constraint tightening and the offline computational cost: the smaller the partitions, the better the description of the uncertainty. The offline computational cost scales as $\mathcal{O}(n_x n_A)$, where n_A is the total number of partitions $\mathbb{A}_m$. A practical way to reduce computational complexity stems from the fact that subsets $\mathbb{A}_m$ need not be of equal size. Thus, one can define an "inner" region, where the subsets are relatively large, thus avoiding numerous unnecessary iterations over subsets that are highly unlikely to be discarded. This approach is particularly useful for speeding up computations. Nevertheless, a drawback of this approach is that the definition of the inner regions is arbitrary. Wrong judgments can lead to large parts of the state-space being discarded, thus making the approach more conservative than it would be if the sets were equally partitioned throughout the state-space.

Remark 5: The sets $\overline{\mathbb{X}}_j$ resulting from Algorithm 1 in [26] may not be convex, even if the starting partitions $\mathbb{A}_m$ are chosen to be convex. Hence, an inner approximation of $\overline{\mathbb{X}}_j$ may be constructed and used instead of the original set. This way, the convexity of the optimal control problem is preserved at the expense of additional conservativeness.

IV. Learning-based SMPC with Chance Constraints for Reference Tracking

This section presents the LB-SMPC problem with chance constraints for system (2) under a stochastic state-dependent uncertainty. To improve control performance, the state-dependent uncertainty model (6) is updated at every sampling time to correct for plant-model mismatch. The chance constraint tightening approach presented in Section III-C is employed to guarantee state constraint satisfaction to a pre-specified probability level, thus ensuring safe system operation. The steady-state parametrization (10) results in an enlargement of the working space (i.e., ROA) of the controller, while circumventing the need to redesign the terminal set every time the reference trajectory is changed.

A. Optimal Control Problem

To steer the system to a desired target, we define the cost function as

$$V_N(\mathbf{c_k}, \tilde{\theta}_k, x_k, x_k^t) = \sum_{j=0}^{N-1} \ell(x_{j|k}, c_{j|k}, \tilde{\theta}_k) + \ell_N(x_{N|k}, \tilde{\theta}_k) + V_o\left(\tilde{x}_k^s - x_k^t\right), \tag{19}$$

where $\mathbf{c}_k := \{c_{0|k}, ..., c_{N-1|k}\}$ is the vector of decision variables (of dimension $n_u N$); ℓ is the stage cost; ℓ_N is the terminal cost; $(\tilde{x}_k^s, \tilde{u}_k^s) = M_\theta \tilde{\theta}_k$ is defined as in (10); and x_k^t is the desired target. To ensure convergence toward x_k^t, an offset cost V_o is added to the cost function. This serves the purpose of penalizing deviations between the artificial reference and the desired target. Here, we choose the stage, terminal, and offset costs according to [29], i.e.,

$$\ell(x_{j|k}, c_{j|k}, \tilde{\theta}_k) = \left|\left|x_{j|k} - x_k^s\right|\right|_Q^2 + \left|\left|u_{j|k} - u_k^s\right|\right|_R^2,$$
$$\ell_N(x_{N|k}, \tilde{\theta}_k) = \left|\left|x_{N|k} - x_k^s\right|\right|_P^2,$$

where P solves the discrete Lyapunov equation (see Appendix C) and V_o has the properties defined in Assumption 3.

Given the measured state x_k at sampling time k, the optimal control problem (OCP) is formulated as

$$V_N^*(x_k, x_k^t) = \min_{\mathbf{c}_k, \tilde{\theta}_k} V_N(\mathbf{c_k}, \tilde{\theta}_k, x_k, x_k^t) \tag{20a}$$
$$\text{s.t. } \tilde{x}_{j+1|k} = A\tilde{x}_{j|k} + B\tilde{u}_{j|k}, \tag{20b}$$
$$x_{j+1|k} = Ax_{j|k} + B\tilde{u}_{j|k} + \mu^{\tilde{w}}(x_{j|k}), \tag{20c}$$
$$\tilde{u}_{j|k} = Kx_{j|k} + L\tilde{\theta}_k + c_{j|k} \tag{20d}$$
$$\tilde{x}_{j|k} \in \overline{\mathbb{X}}_j, \qquad j = 1, \dots, N, \tag{20e}$$
$$\tilde{u}_{j|k} \in \overline{\mathbb{U}}_j, \qquad j = 0, \dots, N-1, \tag{20f}$$
$$\left[\tilde{x}_{N|k}^\top, \tilde{\theta}_k^\top\right]^\top \in \Omega_t^a, \tag{20g}$$
$$\tilde{x}_{0|k} = x_k, \quad x_{0|k} = x_k, \tag{20h}$$

where Ω_t^a is the augmented terminal set, which is defined in (23), and $\mu^{\tilde{w}}(x_{j|k})$ is obtained through (6). Preserving the convexity of the OCP can be achieved by using an inner approximation of the tightened constraints $\overline{\mathbb{X}}_j$ and $\overline{\mathbb{U}}_j$, according to Remark 5. The optimal solution to the OCP is denoted by $\mathbf{c}_k^* := \{c_{0|k}^*, ..., c_{N-1|k}^*\}$. At each sampling time k, the control law (12) takes the form

$$u_k^* = Kx_k + L\tilde{\theta}_k^* + c_{0|k}^* \tag{21}$$

due to the receding-horizon control.

Remark 6: Two models are used in the OCP (20). The first model (20b) represents the nominal predictions. This is also the model used to ensure constraint satisfaction. The second model (20c) represents the predictions of system (2), that is, predictions of the nominal dynamics corrected by the state-dependent uncertainty model (6). Thus, model (20c) is used in the computation of the cost $V_N(\mathbf{c_k}, \tilde{\theta}_k, x_k, x_k^t)$ to improve performance.

Remark 7: The cost $V_N(\mathbf{c_k}, \tilde{\theta}_k, x_k, x_k^t)$ is a function of the states resulting from the learned model (20c), which uses (6) to update the nominal model. Nevertheless, the stability and robustness properties follow from those of the nominal system (see [22] Remark 1).

The choice of the terminal set Ω_t^a is essential for ensuring recursive feasibility of the OCP (20). A necessary condition for recursive feasibility is

$$\left[\tilde{x}_{N|k}, \tilde{\theta}_k\right] \in \Omega_t^a \Rightarrow \left[\tilde{x}_{N|k+1}, \tilde{\theta}_{k+1}\right] \in \Omega_t^a, \quad \forall \tilde{w}(x_k) \in \widetilde{\mathbb{W}}(x_k),$$

Furthermore, the nominal predictions at time $k+1$ are related to those at time k via [29]

$$\begin{bmatrix} \tilde{x}_{N|k+1} \\ \tilde{\theta}_{k+1} \end{bmatrix} = \begin{bmatrix} A_K & BL \\ 0 & I \end{bmatrix} \begin{bmatrix} \tilde{x}_{N|k} \\ \tilde{\theta}_k \end{bmatrix} + \begin{bmatrix} A_K^N \\ 0 \end{bmatrix} \tilde{w}(x_k). \tag{22}$$

Thus, any admissible RPI set for system (22) can be chosen as Ω_t^a, provided that it satisfies the tightened constraints $\tilde{x}_{N|k} \in \overline{\mathbb{X}}_N$ and $K\tilde{x}_{N|k} + L\tilde{\theta}_k \in \overline{\mathbb{U}}_N$.

Assumption 2: The terminal set satisfies

$$\Omega_t^a \subseteq \left\{(x, \theta) : x \in \overline{\mathbb{X}}_N,\ Kx + L\theta \in \overline{\mathbb{U}}_N\right\} \quad \text{and}$$
$$\begin{bmatrix} A_K & BL \\ 0 & I \end{bmatrix} \Omega_t^a \oplus \begin{bmatrix} A_K^N \\ 0 \end{bmatrix} \widetilde{\mathbb{W}}(x) \subseteq \Omega_t^a.$$

Remark 8: Ideally, the terminal set can be defined as the maximal RPI (MRPI) set, which contains all other invariant sets [34]. Specifically, the MRPI set is defined as the infinite set of states and inputs resulting from the nominal system evolution under mode 2, such that the real system does not violate the original constraints. This can be written as

$$\Omega_t^a \subseteq \mathcal{O}_\infty = \left\{x_a : A_a^i x_a \in \overline{\mathbb{X}}_{N+i}^a \quad \forall i \geq 0\right\}, \tag{23}$$

where $A_a = \begin{bmatrix} A_K & BL \\ 0 & I \end{bmatrix}$ and

$$\overline{\mathbb{X}}_{N+i}^a = \left\{x_a = [x, \theta]^\top : x \in \overline{\mathbb{X}}_{N+i},\ Kx + L\theta \in \overline{\mathbb{U}}_{N+i}\right\}.$$

For algorithms to construct such sets, the reader is referred to [43], [44].

B. Stability and Recursive Feasibility

We first introduce the following definitions.

Definition 1: Define the set $\mathcal{R}_j$ as the set of uncertainty values that can be reached in j steps from the origin as

$$\mathcal{R}_j = \bigoplus_{i=0}^{j-1} A_K^i \widetilde{\mathcal{W}}.$$

As $i \to \infty$, $\mathcal{R}_j$ converges to a limit $\mathcal{R}_\infty$ as long as the eigenvalues of A_K are inside the unit disc. $\mathcal{R}_\infty$ is commonly referred to as the minimal robust positively invariant (mRPI) set.

Definition 2: The ROA of the controller is defined as the set of initial states that can be driven to the terminal set within the prediction horizon N, while adhering to the constraints for all possible uncertainty realizations

$$\mathbb{X}_{\text{ROA}} = \left\{ x_k \in \mathbb{R}^{n_x} : \exists\ \mathbf{c}_k, \tilde{\theta}_k \text{ s.t. (20) is feasible} \right\}.$$

To prove the recursive feasibility, stability, and convergence properties of the LB-SMPC strategy for trajectory tracking, the following assumptions are made.

Assumption 3:

i) $\Psi = \Psi^\top \succ 0$ is a symmetric positive definite matrix.
ii) The eigenvalues of $A_K = A + BK$ are inside the unit disk.
iii) The set Ω_t^a is non-empty.
iv) The offset cost $V_o(\cdot) : \mathbb{R}^{n_x} \to \mathbb{R}$ is positive definite, convex, sub-differential function with $V_o(0) = 0$.

Theorem 1 summarizes the recursive feasibility, stability, and convergence properties of the LB-SMPC.

Theorem 1: If Assumptions 1-3 hold and the target x_k^t is asymptotically constant, the closed-loop system

$$x_{k+1} = A_K x_k + B\left(L\tilde{\theta}_k^* + c_{0|k}^*\right) + \tilde{v}(x_k) \qquad (24)$$

under the control law (21) satisfies:

i) For all targets x_k^t and initial states $x \in \mathbb{X}_{\text{ROA}}$, the evolution of system (24) is robustly feasible. That is, $x_k \in \mathbb{X}_{\text{ROA}},\ \forall k \geq 1$. Furthermore, system (24) satisfies chance constraints (9) and hard input constraints (8b) for all admissible uncertainties.
ii) $\lim_{k\to\infty} c_{0|k}^*(x_k, x_k^t) = 0$.
iii) Assuming that the asymptotic target is reachable, i.e., $x_\infty^t \in \overline{\mathbb{X}}_t$, then the controlled variable x_k converges to the set $\{x_\infty^t\} \oplus R_\infty$ with probability one.
iv) Assuming that the asymptotic target is not reachable, i.e., $x_\infty^t \notin \overline{\mathbb{X}}_t$, then the controlled variable x_k converges to the set $\{\tilde{x}_s\} \oplus R_\infty$ with probability one, where $\tilde{x}_s$ is the reachable nominal steady state controlled variable that minimizes the offset cost. That is,

$$\tilde{x}_s = \arg\min_{x \in \overline{\mathbb{X}}_t} V_o\left(x - x_\infty^t\right).$$

Proof: See Appendix C. ∎

V. Results and Discussion: Application to APPJ

This section presents the results of closed-loop simulations and real-time control experiments of the LB-SMPC applied to the APPJ in Fig. 1. We first discuss the constraint tightening approach of Section III-C, followed by the construction of the terminal set for tracking. Next, Monte Carlo simulations of the closed-loop system are presented to discuss the chance constraint handling, followed by the real-time control experiments. Lastly, online training of the GP model of the system uncertainty is discussed to mimic a situation in which no data on plant-model mismatch is available a priori, for example, in plasma treatments of a new patient or even on different tissues on the same patient.

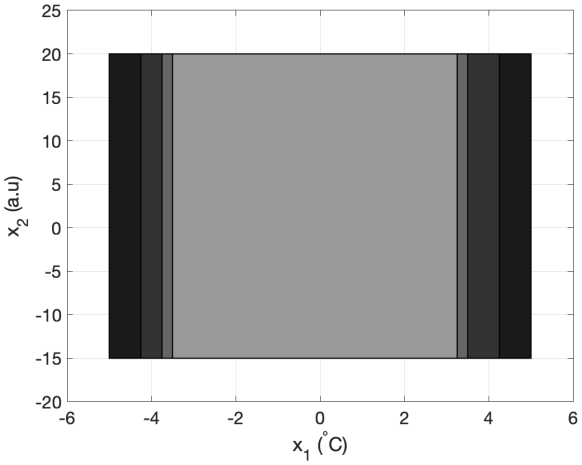

Fig. 3: Recursively tightened constraint sets. Darkest shade represents the original constraints that progressively become lighter with increasing tightening steps.

The LB-SMPC controller is implemented in Python using the CasADi toolbox for nonlinear optimization [45]. Data acquisition is carried out in Python, while polyhedron set operations for constraint tightening are performed offline in MATLAB using the MPT3 toolbox [42] and the invariant set toolbox [44]. In the APPJ testbed, measurements become available every 1.3 s. Thus, we balance computational tractability with prediction quality of the GP by choosing $N = 4$ and $\ell = 1$ with 200 input-output samples for offline training of the GP model (6).

A. Design of Tightened Constraints and Terminal Set

Due to the close relationship of surface temperature and thermal dose accumulation on the target surface [10], we aim to achieve guaranteed satisfaction of the surface temperature constraints. We define the deviation variables $x_1 = T - T^s$, $x_2 = I - I^s$, $u_1 = q - q^s$, and $u_2 = P - P^s$. We consider the following constraints for the APPJ

$$\mathbb{P}[x_1 \leq 5] \geq 1 - \varepsilon, \quad \mathbb{P}[-x_1 \leq 5] \geq 1 - \varepsilon, \qquad (25a)$$
$$x \in \mathbb{X} = \{x \mid -15 \leq x_2 \leq 20\}, \qquad (25b)$$
$$u \in \mathbb{U} = \{u \mid -1.5 \leq u_1 \leq 7,\ -2.5 \leq u_2 \leq 2\}, \qquad (25c)$$

where x_1 is measured in °C, x_2 in a.u., u_1 in slm, and u_2 in W. Here, we choose $\varepsilon = 0.2$. State-dependent chance constraint tightening of the temperature constraints ($x_1 = T - T^s$) is carried out offline, as described in Section III-C. First, we use data of the plant-model mismatch y^{mis} in (3) to obtain the ECDF of $\tilde{v}(x)$ and accordingly determine $\hat{\gamma}_i$.[2] Then, we iterate over partitions $\mathbb{A}_m$ and follow Algorithm 1 in [26] to choose whether to keep or discard each $\mathbb{A}_m$. The final tightened and convexified constraints are shown in Fig.

[2]Alternatively, one could use the learned model of $\tilde{w}(x)$ in (6) to analytically obtain the CDF of $\tilde{w}(x)$. These two approaches should, in principle, provide equivalent results.

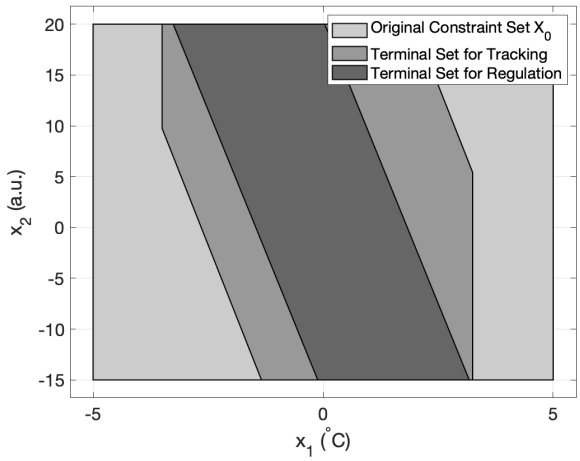

Fig. 4: Terminal set for tracking (gray) compared to that for regulation to a fixed target (dark gray). The original constraint set $\mathbb{X}_0$ is shown in light gray.

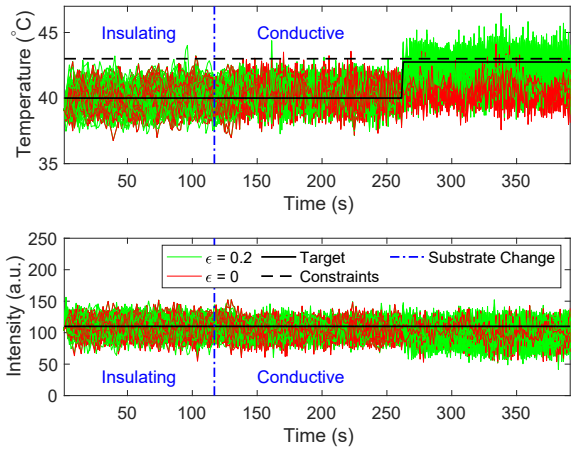

Fig. 5: Closed-loop state profiles for LB-SMPC under 100 different uncertainty realizations. Green profiles correspond to 20% allowed constraint violation ($\epsilon = 0.2$), whereas red profiles correspond to 0% allowed constraint violation ($\epsilon = 0$). The plasma treatment undergoes a drastic change at $t = 120$ s by changing the surface from insulating to conductive.

3. The darkest shade represents $k = 0$, whereas the lightest $k = 3$.

Next, we compute the terminal set to ensure stability and recursive feasibility of the LB-SMPC for reference tracking; see Section IV. Fig. 4 shows the projection of the terminal set for tracking onto the state-space in comparison with the terminal set for regulation to a fixed target. Clearly, the terminal set for tracking is larger than the one for regulation, providing additional feasibility for tracking different targets without the need to redesign the terminal set. In addition, this added flexibility enlarges the domain of attraction of the controller, as discussed in the next subsection.

B. Closed-loop Simulations

One of the main objectives of this work is to enable predictive control of plasma treatment of surfaces with different electrical and thermal properties; for example, in plasma treatment of tissues with spatially varying properties. To mimic such a scenario, plasma operation begins by tracking a target ($T^t = 40$ °C, $I^t = 110$ a.u.) on an insulating surface, which is the "expected" operating surface. That is, the insulating surface is the one on which the linear state-space model (1) has been identified. At $t = 120$ s, the surface is changed to a conductive surface through automated movement of the jet away from the cover slip. In addition, to test whether the controller can guarantee state chance constraint satisfaction, a target change on the conductive surface is introduced at $t = 260$ s to drive the temperature close to its upper constraint ($T^t = 42.5$ °C, $I^t = 110$ a.u.). The intensity target is kept constant and far enough from its constraint.

Fig. 5 illustrates 100 Monte Carlo (MC) runs for two probabilities of constraint violation, i.e., 20% (green) and 0% (red). In both cases, the target $T^t = 40$ °C is tracked fairly well on the insulating as well as the conductive surface, as shown by the the root mean squared errors (RMSEs) in Table I. This implies that the GP model of the state-dependent uncertainty $\tilde{v}(x)$ can adequately predict the plant-model mismatch arising from a change in the surface. As a result, both controllers yield good control performance up until the target is changed.

However, once the target is changed from $T^t = 40$ °C to $T^t = 42$ °C at time $t = 260$ s, the controller with $\varepsilon = 0$ undershoots the target on average, but does not violate the temperature constraint, except for very few time instances. This occurs because we have truncated the distribution of $\tilde{w}(x)$ to lie in $\widetilde{\mathbb{W}}(x)$, which has a finite support according to (7) based on the 99% confidence interval of the GP prediction. Consequently, in 1% of the MC simulations, there might be an uncertainty realization that drives the system outside of the constraints. Table I also indicate a larger error in the trajectory of x_1 for the controller $\varepsilon = 0$ compared to the controller with $\varepsilon = 0.2$. Note that the RMSEs for x_2 are fairly similar because there is no operation close to the constraints. As expected, increasing the value of ε to 0.2 results in a better tracking performance on average at the expense of violating the constraint. The observed level of constraint violation (Table I) is 19.1%, which is close to the specified upper bound of 20%. This suggests that the chance

TABLE I: Results of 100 closed-loop simulations of LB-SMPC for two levels of acceptable constraint violation, ε, in the chance constraint.

	$\varepsilon = 0$		$\varepsilon = 0.2$	
	T (°C)	I (a.u.)	T (°C)	I (a.u.)
RMSE	1.4	1.4	1.0	1.5
Constraint Violation	$\leq 1\%$	-	19.1%	-

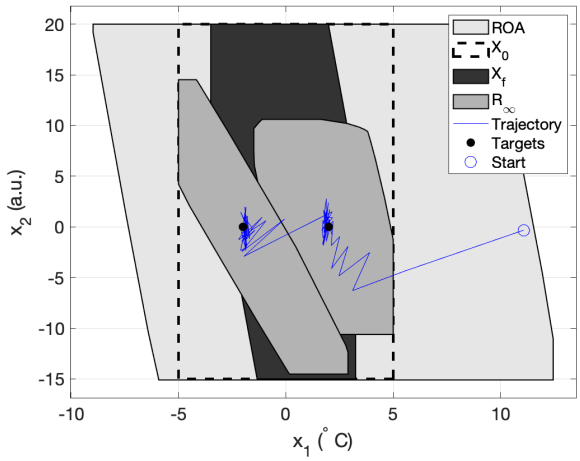

Fig. 6: Phase plot of the average of 50 Monte Carlo state profiles (blue line) starting at the edge of the region of attraction (ROA) and tracking two reference targets (black dots). The light gray area illustrates the enlarged ROA, resulting from the SMPC with chance constraints for reference tracking. Constraints are shown as dotted lines ($\mathbb{X}_0$). mRPI sets are shaded in darker gray ($\mathcal{R}_\infty$). The terminal set is colored in charcoal gray ($\mathbb{X}_f$).

constraint tightening is not conservative.

Fig. 6 shows a phase plot with the the average trajectory of the LB-SMPC with $\varepsilon = 0.2$ over 50 MC simulations. The LB-SMPC for tracking can smoothly handle target changes while also enlarging the ROA. Since the chance-constrained formulation of the LB-SMPC allows for some constraint violation, the controller can start from a point outside the original constraint set and still converge to the terminal set while staying below the constraint violation threshold. In contrast, this is not allowed when $\varepsilon = 0$. In the latter case, any starting point outside the original constraint is by definition considered infeasible for the OCP. In addition, the enlarged terminal set for tracking compared to that for regulation allows further increase in the size of the ROA, since there are more allowable points to which the controller has to drive the system at the end of the prediction horizon.

As shown in Fig. 6, after each target change, the system converges to its respective mRPI set, while the closed-loop mean (blue) converges to the desired target. Due to the nonlinear nature of the APPJ system (2), the mRPI sets change in shape and size. This occurs because the linear system is identified around one operating point. Therefore switching to another operating point would result in a different linear system identification.

C. Real-time Control Experiments

1) Offset-Free MPC versus LB-MPC: The performance of learning-based MPC using GP regression (LB-MPC) is compared to that of offset-free MPC (OF-MPC) [46] via real-time control experiments on the APPJ testbed. Note that the constraints in LPB-MPC are not tightened. The state and

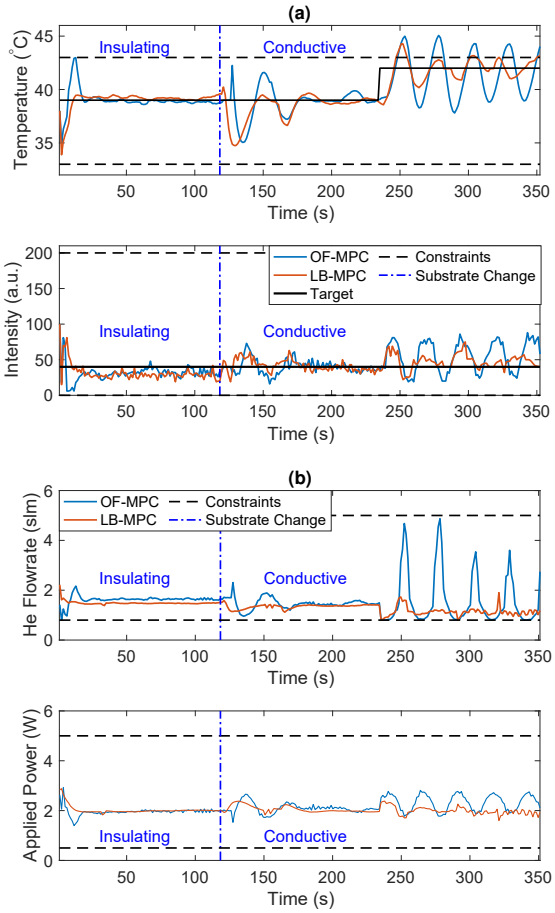

Fig. 7: Real-time control experiments comparing the performance of offset-free MPC (OF-MPC) with learning-based MPC (LB-MPC), i.e., without constraint tightening. (a) State profiles. (b) Input profiles. The plasma treatment undergoes a drastic change at $t = 120$ s by changing the surface from insulating to conductive. At $t = 235$ s, the target for surface temperature T changes from 39 °C to 43 °C.

input profiles are shown in Fig. 7. The performance of OF-MPC on the insulating surface is comparable to that of LB-MPC, with the exception of an initial overshoot. At $t = 120$ s, when the surface is changed from insulating to conductive, the OF-MPC exhibits a more-oscillatory behavior, while the LB-MPC is able to settle to the desired target relatively faster. This is because the OF-MPC is estimating the plant-model mismatch and then correcting the target accordingly. However, because of the plant-model mismatch, the OF-MPC can overcompensate. That is, once the OF-MPC has determined the error and calculated the artificial target, it takes action based on the erroneous model and consequently overshoots the desired target. This, in turn, leads to a new estimation of the error in the opposite direction, and so on. This is exacerbated after the surface change from insulating to conductive, when the OF-MPC controller is driven at the verge of instability.

In contrast, the LB-MPC uses the current state (and

TABLE II: Closed-loop performance of OF-MPC, LB-MPC (no constraint tightening), and LB-SMPC with $\varepsilon = 0$, quantified through the root mean squared error (RMSE) and constraint violation fraction in real-time control experiments.

	OF-MPC		LB-MPC		LB-SMPC with $\varepsilon = 0$	
	T / °C	I / a.u.	T / °C	I / a.u.	T / °C	I / a.u.
RMSE	1.6	18.2	1.2	12.2	0.8	9.9
Constraint Violation	10%	-	5%	-	0%	-

previous error) to predict the GP correction term in the next step, i.e., $\tilde{w}(x)$ in (6). Therefore, the first control input is chosen with a partially corrected model, which leads to smoother and less oscillatory behavior. The accuracy of the prediction is heavily dependent on the quantity and quality of the data used for training the GP model. The performance of both controllers is summarized in Table II. In spite of the better performance of LB-MPC compared to OF-MPC, the constraint is still violated in both cases shortly after the target is changed.

2) LB-MPC versus LB-SMPC: In real-time control experiments, we now compare the performance of LB-MPC to the proposed LB-SMPC with $\epsilon = 0$ in the chance constraint. The state and input profiles are shown in Fig. 8. The two controllers exhibit comparable performance on the insulating surface. In the region of operation close to the temperature constraint, however, the LB-MPC exhibits damped oscillations around the desired target, whereas the LB-SMPC overshoots it once and then consistently remains below it. This happens because the constraints are tightened in such a way that the target $T^t = 42$ °C cannot be reached. Instead, the LB-SMPC settles to a value just below the target, which corresponds to the maximum temperature allowable by the tightened constraint design. This illustrates the trade-off between controller performance and robustness. More importantly, in case of the LB-SMPC, the temperature constraint is not violated since ϵ was set to 0. This is particularly important in safety-critical applications of APPJ, where exceeding the specified limitations may harm the patient and reduce the therapeutic effects of the plasma.

D. Online versus Offline Model Learning

Thus far, we have used data collected from plasma treatment of insulating and conductive surfaces to train the GP model (5). In practice, however, the user may not know a priori that there will be a change in surface properties. Here, we compare online versus offline training of the GP model (5). We opt to track a surface temperature target far from the constraint when there is a transition from treating an insulating surface to a conductive surface. We use 30 training samples to train the GP model, since this allows retraining the model online within the specified sampling time of 1.3 s. We continuously update the dataset to include the sample collected from the latest time step, while eliminating the sample collected in the oldest time step, to keep the size of the datasets constant. The data structure is shown in Appendix D.

Fig. 9 shows the closed-loop surface temperature profile for the LB-MPC controllers when the GP model (5) is trained offline (red) and online (blue). Since a limited dataset is used for training of the GP model (30 samples), the LB-MPC trained offline exhibits an offset in tracking the target. This is expected since there is no offset-free formulation enabling offset elimination, while the predictions that are supposed to correct the nominal model are inaccurate. On the other hand, the LB-MPC trained online gradually learns the plant-model mismatch as it collects more samples relevant to its region of operation. Even though those samples are no longer relevant

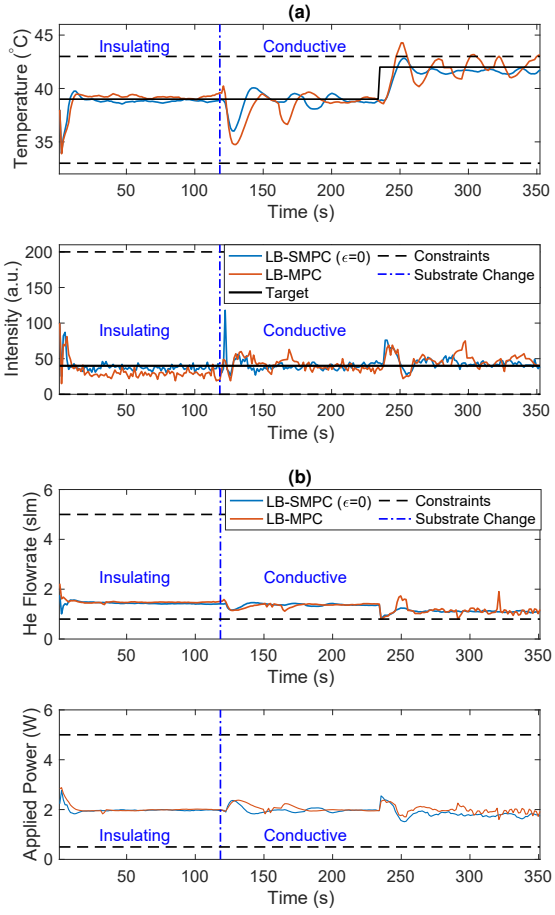

Fig. 8: Real-time control experiments comparing the performance of learning-based MPC (LB-MPC), i.e., without constraint tightening, with learning-based stochastic MPC (LB-SMPC), i.e., with $\varepsilon = 0$. (a) State profiles. (b) Input profiles. The plasma treatment undergoes a drastic change at $t = 120$ s by changing the surface from insulating to conductive. At $t = 235$ s, the target for surface temperature T changes from 39 °C to 43 °C.

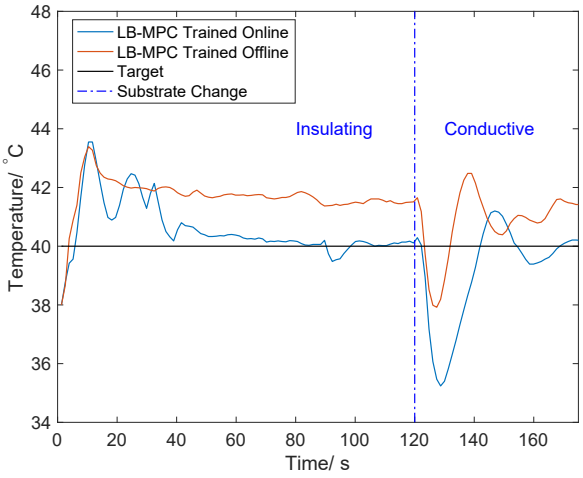

Fig. 9: Closed-loop surface temperature under LB-MPC for the cases in which the Gaussian process model that corrects for plant-model mismatch is trained online (blue) and offline (red). The plasma treatment undergoes a drastic change at $t = 120$ s by changing the surface from insulating to conductive.

after the surface is changed at $t = 120$ s, the controller keeps adding training points collected from the metal surface and once again manages to drive the system close to the desired target.

The RMSE of the LB-MPC trained online is 1.2 °C, whereas the RMSE of the LB-MPC trained offline is 1.6 °C. Although the LB-MPC trained online does consistently better, it exhibits a much larger undershoot when the surface is changed due to the specialized nature of the training points involved, i.e., the training points are all from the insulating surface, so they are no longer applicable to the conductive surface. The LB-MPC trained offline uses samples from different surfaces, whereas the controller trained online only relies on samples pertinent to operation on the insulating surface at $t = 120$ s. Thus, when the surface transition occurs, the LB-MPC trained online cannot quickly accommodate the surface change, but eventually adapts and reaches the desired target.

VI. Conclusions and Future Work

This paper presents a learning-based stochastic MPC strategy for reference tracking. The proposed formulation is experimentally tested on an atmospheric pressure plasma jet with prototypical applications in plasma medicine. The promise of the LB-SMPC approach for plasma jets is demonstrated by using Gaussian process regression to capture the state-dependent plant-model mismatch that results from plasma treatment of surfaces with vastly different electrical and thermal properties. Closed-loop simulations and real-time control experiments indicate the superior performance of LB-SMPC to that of offset-free MPC and LB-MPC with no robustness guarantees. Future work will investigate the extension of the proposed LB-SMPC strategy to the output feedback case. In addition, online adaptation of the tightened constraint sets will be investigated to fully leverage the learning of a state-dependent uncertainty for learning-based control with safety guarantees.

References

[1] J. Jeong, S. Babayan, V. Tu, J. Park, I. Henins, R. Hicks, and G. Selwyn, "Etching materials with an atmospheric-pressure plasma jet," *Plasma Sources Science and Technology*, vol. 7, no. 3, p. 282, 1998.

[2] J. Heinlin, G. Isbary, W. Stolz, G. Morfill, M. Landthaler, T. Shimizu, B. Steffes, T. Nosenko, J. Zimmermann, and S. Karrer, "Plasma applications in medicine with a special focus on dermatology," *Journal of the European Academy of Dermatology and Venereology*, vol. 25, no. 1, pp. 1–11, 2011.

[3] H.-R. Metelmann, D. S. Nedrelow, C. Seebauer, M. Schuster, T. von Woedtke, K. D. Weltmann, S. Kindler, P. H. Metelmann, S. E. Finkelstein, D. D. Von Hoff, and F. Podmelle, "Head and neck cancer treatment and physical plasma," *Clinical Plasma Medicine*, vol. 3, no. 1, pp. 17–23, 2015.

[4] G. Fridman, G. Friedman, A. Gutsol, A. B. Shekhter, V. N. Vasilets, and A. Fridman, "Applied plasma medicine," *Plasma Processes and Polymers*, vol. 5, no. 6, pp. 503–533, 2008.

[5] D. Gidon, B. Curtis, J. A. Paulson, D. B. Graves, and A. Mesbah, "Model-Based Feedback Control of a kHz-Excited Atmospheric Pressure Plasma Jet," *IEEE Transactions on Radiation and Plasma Medical Sciences*, vol. 2, no. 2, pp. 129–137, 2018.

[6] M. Laroussi, M. Kong, G. Morfill, and W. Stolz, *Plasma medicine: applications of low-temperature gas plasmas in medicine and biology*. Cambridge University Press, 2012.

[7] K. D. Weltmann and T. von Woedtke, "Plasma medicine – current state of research and medical application," *Plasma Physics and Controlled Fusion*, vol. 59, no. 1, p. 014031, 2016.

[8] J. Shin and L. L. Raja, "Run-to-run variations, asymmetric pulses, and long time-scale transient phenomena in dielectric-barrier atmospheric pressure glow discharges," *Journal of Physics D: Applied Physics*, vol. 40, no. 10, pp. 3145–3154, 2007.

[9] M. Dünnbier, A. Schmidt-Bleker, J. Winter, M. Wolfram, R. Hippler, K. D. Weltmann, and S. Reuter, "Ambient air particle transport into the effluent of a cold atmospheric-pressure argon plasma jet investigated by molecular beam mass spectrometry," *Journal of Physics D: Applied Physics*, vol. 46, no. 43, p. 435203, 2013.

[10] D. Gidon, D. B. Graves, and A. Mesbah, "Effective dose delivery in atmospheric pressure plasma jets for plasma medicine: a model predictive control approach," *Plasma Sources Science and Technology*, vol. 26, no. 8, p. 085005, 2017.

[11] D. Gidon, D. B. Graves, and A. Mesbah, "Predictive control of 2D spatial thermal dose delivery in atmospheric pressure plasma jets," *Plasma Sources Science and Technology*, vol. 28, no. 8, p. 085001, 2019.

[12] J. Rawlings, D. Mayne, and M. Diehl, *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing, 2017.

[13] S. Wu, Z. Wand, Q. Huang, X. Lu, and Y. Pan, "Study on a room-temperature air plasma for biomedical application," *IEEE Transactions on Plasma Science*, vol. 39, no. 6, pp. 1489–1495, 2011.

[14] A. Mesbah and D. B. Graves, "Machine learning for modeling, diagnostics, and control of non-equilibrium plasmas," *Journal of Physics D: Applied Physics*, vol. 52, no. 30, p. 30LT02, 2019.

[15] D. Breden and L. L. Raja, "Computational study of the interaction of cold atmospheric helium plasma jets with surfaces," *Plasma Sources Science and Technology*, vol. 23, no. 6, p. 065020, 2014.

[16] L. Ji, W. Yan, Y. Xia, and D. Liu, "The effect of target materials on the propagation of atmospheric-pressure plasma jets," *Journal of Applied Physics*, vol. 123, no. 18, p. 183302, 2018.

[17] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin, "A general safety framework for learning-based control in uncertain robotic systems," *IEEE Transactions on Automatic Control*, 2018.

[18] S. Arimoto, S. Kawamura, and F. Miyazaki, "Bettering operation of robots by learning," *Journal of Robotic systems*, vol. 1, no. 2, pp. 123–140, 1984.

[19] J. Kocijan, R. Murray-Smith, C. E. Rasmussen, and A. Girard, "Gaussian Process Model Based Predictive Control," in *Proceedings of the 2004 American Control Conference*, 2004, pp. 2214–2219.

[20] B. Likar and J. Kocijan, "Predictive control of a gas–liquid separation plant based on a gaussian process model," *Computers & Chemical Engineering*, vol. 31, no. 3, pp. 142–152, 2007.
[21] C. J. Ostafew, A. P. Schoellig, and T. D. Barfoot, "Learning-based nonlinear model predictive control to improve vision-based mobile robot path-tracking in challenging outdoor environments," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2014, pp. 4029–4036.
[22] A. Aswani, H. Gonzalez, S. S. Sastry, and C. Tomlin, "Provably safe and robust learning-based model predictive control," *Automatica*, vol. 49, no. 5, pp. 1216–1226, 2013.
[23] D. Limon, J. Calliess, and J. M. Maciejowski, "Learning-based nonlinear model predictive control," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 7769–7776, 2017.
[24] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. MIT Press, 2006.
[25] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, "Learning-based model predictive control: Toward safe learning in control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, pp. 10.1–10.28, 2019.
[26] R. Soloperto, M. A. Müller, S. Trimpe, and F. Allgöwer, "Learning-based robust model predictive control with state-dependent uncertainty," *IFAC-PapersOnLine*, vol. 51, no. 20, pp. 442–447, 2018.
[27] L. Hewing, J. Kabzan, and M. N. Zeilinger, "Cautious model predictive control using gaussian process regression," *IEEE Transactions on Control Systems Technology*, 2019.
[28] A. D. Bonzanini, J. A. Paulson, and A. Mesbah, "Safe learning-based model predictive control under state-and input-dependent uncertainty using scenario trees," in *Proceedings of the IEEE Conference on Decision and Control. Jeju Island, Republic of Korea. Accepted*, 2020.
[29] J. Paulson, T. Santos, and A. Mesbah, "Mixed stochastic-deterministic tube mpc for offset-free tracking in the presence of plant-model mismatch," *Journal of Process Control*, vol. 83, pp. 102–120, 2019.
[30] D. Breden, K. Miki, and L. Raja, "Self-consistent two-dimensional modeling of cold atmospheric-pressure plasma jets/bullets," *Plasma Sources Science and Technology*, vol. 21, no. 3, p. 034011, 2012.
[31] P. Van Overschee and B. De Moor, *Subspace identification for linear systems: TheoryImplementationApplications*. Springer Science & Business Media, 2012.
[32] J. Umlauft, T. Beckers, and S. Hirche, "Scenario-based optimal control for gaussian process state space models," in *Proceedings of the European Control Conference*, 2018, pp. 1386–1392.
[33] L. Hewing, E. Arcari, L. P. Fröhlich, and M. N. Zeilinger, "On simulation and trajectory prediction with gaussian process dynamics," in *Proceedings of Machine Learning Research*, 2020, pp. 424–434.
[34] D. Limón, I. Alvarado, T. Alamo, and E. F. Camacho, "MPC for tracking piecewise constant references for constrained linear systems," *Automatica*, vol. 44, pp. 2382–2387, 2008.
[35] M. Balandat, "Constrained robust optimal trajectory tracking: Model predictive control approaches," in *Control Systems Technology*. Darmstadt University of Technology, 2010.
[36] H. Michalska and D. Q. Mayne, "Robust receding horizon control of constrained nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 38, pp. 1623–1633, 1993.
[37] B. Kouvaritakis, M. Cannon, S. V. Rakovic, and Q. Cheng, "Explicit use of probabilistic distributions in linear predictive control," 2010.
[38] L. Chisci, J. A. Rossiter, and G. Zappa, "Systems with persistent disturbances: predictive control with restricted constraints," *Automatica*, vol. 37, pp. 1019–1028, 2001.
[39] T. A. N. Heirung, J. A. Paulson, J. O'Leary, and A. Mesbah, "Stochastic model predictive control–how does it work?" *Computers & Chemical Engineering*, pp. 158–170, 2018.
[40] T. L. M. Santos, A. D. Bonzanini, and A. Mesbah, "A constraint-tightening approach to nonlinear model predictive control with chance constraints for stochastic systems," in *Proceedings of the American Control Conference*, Philadelphia, PA, USA, 2019, submitted.
[41] D. Azriel and A. Schwartzman, "The empirical distribution of a large number of correlated normal variables," *Journal of the American Statistical Association*, vol. 110, pp. 1217–1228, 2015.
[42] M. Herceg, M. Kvasnica, C. Jones, and M. Morari, "Multi-Parametric Toolbox 3.0," in *Proceedings of the European Control Conference*, Zürich, Switzerland, 2013, pp. 502–510.
[43] I. Kolmanovsky and E. G. Gilbert, "Theory and computation of disturbance invariant sets for discrete-time linear systems," *Mathematical Problems in Engineering*, vol. 4, no. 4, pp. 317–367, 1998.
[44] E. C. Kerrigan, "Robust constraint satisfaction: Invariant sets and predictive control," Ph.D. dissertation, University of Cambridge, 2001.
[45] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.
[46] G. Pannocchia and J. B. Rawlings, "Disturbance models for offset-free model-predictive control," *AIChE journal*, vol. 49, no. 2, pp. 426–437, 2003.
[47] A. Ferramosca, D. Limon, A. H. González, I. Alvarado, and E. F. Camacho, "Robust mpc for tracking zone regions based on nominal predictions," *Journal of Process Control*, vol. 22, no. 10, pp. 1966–1974, 2012.

APPENDIX

A. APPJ Model Parameters

The APPJ model is summarized in Table III. The validation data for the GP model is shown in Fig. 10.

B. Proof of Theorem 1

1) Proof of Recursive Feasibility: Recursive feasibility is proven through a similar procedure as in [29]. That is, we seek to demonstrate that, given a solution at time k, an explicit candidate solution will satisfy the constraints at time $k+1$. Note that throughout the proof, we assume that $|\tilde{v}(x)| \leq |\tilde{w}(x)| \in \widetilde{\mathbb{W}}(x)$, i.e., the state-dependent uncertainty model $\tilde{w}(x)$ is at least as large as the real underlying uncertainty.

To begin with, recall that the arguments of the minimization of the OCP (20) at time k are denoted by $c_k^* = [c_{0|k}^*, c_{1|k}^*, \ldots, c_{N-1|k}^*]$ and $\tilde{\theta}_k^*$. The nominal state predictions are then given by

$$\tilde{x}_{j+1|k}^* = A_K \tilde{x}_{j|k}^* + BL\tilde{\theta}_k^* + Bc_{j|k}^*, \quad \tilde{x}_{0|k}^* = x_k.$$

The candidate solutions at time $k+1$ are defined as $c_{k+1}^c = [c_{1|k}^*, c_{2|k}^*, \ldots, c_{N-1|k}^*, 0]$ and $\tilde{\theta}_{k+1}^c = \tilde{\theta}_k^*$. In addition, we denote the nominal state predictions corresponding to the initial state $\tilde{x}_{0|k+1}^c = x_{k+1}$ as $\tilde{x}_{j|k+1}^c,\ j \geq 0$.

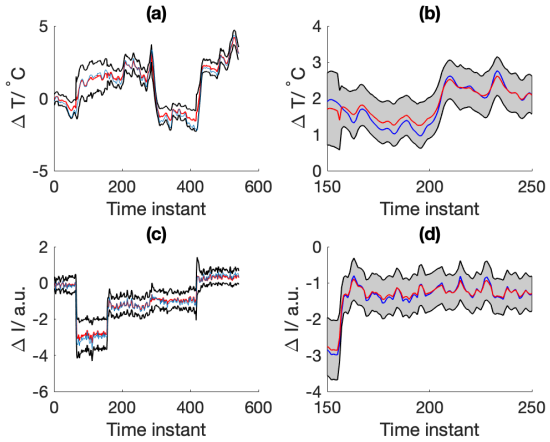

Fig. 10: Validation Data. (a) Temperature correction validation; (b) Inset of (a); (c) Intensity correction validation; (d) Inset of (c). Predictions are shown in red and real data is shown in blue. The 99% confidence interval is shaded in light gray.

TABLE III: Model and MPC Parameters

Problem characteristic	Specification
Deviation variables	$y = \begin{bmatrix} T - 38\ ^\circ\text{C} \\ 0.1(I - 100)\ \text{a.u.} \end{bmatrix},\ u = \begin{bmatrix} q - 3.0\ \text{slm} \\ P - 3.0\ \text{W} \end{bmatrix}$
APPJ Model	$A = \begin{bmatrix} 0.9418 & 0.0821 \\ -0.0552 & 0.0080 \end{bmatrix},\ B = \begin{bmatrix} -0.2970 & -0.0180 \\ 1.6664 & 3.3180 \end{bmatrix},\ C = I_2,\quad d(x) \leftarrow$ from GP
Hard constraints	$x_1 \in [-5, 5],\ x_2 \in [-15, 20]\ u_1 \in [-1.5, 7],\ u_2 \in [-2.5, 2]$
Time-related parameters	$N = 4,\ l = 1$
Training Data	$N_{\text{offline training}} = 200,\quad N_{\text{online training}} = 30$

Having established these definitions and notational conventions, we can proceed to relate $\tilde{x}^c_{0|k+1}$ to $\tilde{x}^*_{1|k}$. Using (2) and (6), we can write

$$\tilde{x}^c_{0|k+1} = x_{k+1} = Ax_k + Bu_k + \tilde{w}(x_k).$$

Substituting $u_k = \tilde{u}^*_{0|k} = Kx_k + L\tilde{\theta}^*_k + c^*_{0|k}$, we obtain

$$\begin{aligned}\tilde{x}^c_{0|k+1} &= Ax_k + B\left(Kx_k + L\tilde{\theta}^*_k + c^*_{0|k}\right) + \tilde{w}(x_k) = \\ &= (A + BK)x_k + BL\tilde{\theta}^*_k + Bc^*_{0|k} + \tilde{w}(x_k) = \\ &= A_K x_k + BL\tilde{\theta}^*_k + Bc^*_{0|k} + \tilde{w}(x_k) = \\ &= \tilde{x}^*_{1|k} + \tilde{w}(x_k).\end{aligned}$$

By definition, $\tilde{x}^c_{1|k+1} = A_K\tilde{x}^c_{0|k+1} + BL\tilde{\theta}^c_{k+1} + Bc^c_{0|k+1}$. However, from the control sequences, it is apparent that $c^c_{0|k+1} = c^*_{1|k}$. In addition, as it has already been pointed out, $\tilde{\theta}^c_{k+1} = \tilde{\theta}^*_k$. Thus, making these substitutions yields

$$\begin{aligned}\tilde{x}^c_{1|k+1} &= A_K\left(\tilde{x}^*_{1|k} + \tilde{w}(x_k)\right) + BL\tilde{\theta}^*_k + Bc^*_{1|k} = \\ &= \left(A_K\tilde{x}^*_{1|k} + BL\tilde{\theta}^*_k + Bc^*_{1|k}\right) + A_K\tilde{w}(x_k) = \\ &= \tilde{x}^*_{2|k} + A_K\tilde{w}(x_k).\end{aligned}$$

We can generalize the above by induction to obtain

$$\tilde{x}^c_{j|k+1} = \tilde{x}^*_{j+1|k} + A^j_K\tilde{w}(x_k), \quad \forall j \in \mathbb{N}_{[0,N-1]}. \tag{26}$$

Similarly, the candidate input can be re-written as

$$\begin{aligned}\tilde{u}^c_{j|k+1} &= K\tilde{x}^c_{j|k+1} + L\tilde{\theta}^c_{k+1} + c^c_{j|k+1} \\ &= K\left(\tilde{x}^*_{j+1|k} + A^j_K\tilde{w}(x_k)\right) + L\tilde{\theta}^*_k + c^*_{j+1|k} = \\ &= \left(K\tilde{x}^*_{j+1|k} + L\tilde{\theta}^*_k + c^*_{j+1|k}\right) + KA^j_K\tilde{w}(x_k),\end{aligned}$$

which yields

$$\tilde{u}^c_{j|k+1} = \tilde{u}^*_{j+1|k} + KA^j_K\tilde{w}(x_k), \quad \forall j \in \mathbb{N}_{[0,N-1]}. \tag{27}$$

At this point, we will focus on two separate cases: (i) when $1 \leq j < N$, and (ii) when $j = N$.

i) $1 \leq j < N$.
From (26) and (27), we can set $j \leftarrow j - 1$ to obtain

$$\tilde{x}^c_{j-1|k+1} = \tilde{x}^*_{j|k} + A^{j-1}_K\tilde{w}(x_k), \tag{28}$$

$$\tilde{u}^c_{j-1|k+1} = \tilde{u}^*_{j|k} + KA^{j-1}_K\tilde{w}(x_k). \tag{29}$$

By construction of the sets $\overline{\mathbb{X}}_j$ (see Algorithm 1 in [26]), we know that for a region $\mathbb{A}_m$

$$\tilde{x}_{j|k} \in \mathbb{A}_m \subseteq \overline{\mathbb{X}}_j \Rightarrow \mathbb{A}_m \oplus A^{j-1}_K\hat{\mathbb{W}}_{-j}(\mathbb{A}_m) \subseteq \overline{\mathbb{X}}_{j-1}, \tag{30}$$

where $\hat{\mathbb{W}}_{-j}(\mathbb{A}_m) := \bigcup_{x\in\mathbb{H}_{-j}(\mathbb{A}_m)} \widetilde{\mathbb{W}}(x)$.

Recall that the uncertainty set $\widetilde{\mathbb{W}}(x_k)$ can be overestimated by

$$\widetilde{\mathbb{W}}(x_k) \subseteq \hat{\mathbb{W}}_{-j}(\mathbb{A}_m),$$

which means that $\tilde{w}(x_k) \in \hat{\mathbb{W}}_{-j}(\mathbb{A}_m)$. This, in combination with (28) and (30), leads to the conclusion that

$$\tilde{x}^*_{j|k} \in \overline{\mathbb{X}}_j \Rightarrow \tilde{x}^c_{j-1|k+1} \in \overline{\mathbb{X}}_{j-1}, \quad \forall j \in \mathbb{N}_{0,N-1},$$

and, similarly,

$$\tilde{u}^*_{j|k} \in \overline{\mathbb{U}}_j \Rightarrow \tilde{u}^c_{j-1|k+1} \in \overline{\mathbb{U}}_{j-1}, \quad \forall j \in \mathbb{N}_{0,N-2}.$$

ii) $j = N$. Here, we show that the terminal constraint is respected at time $k+1$. We first show that at $j = N$

$$\begin{aligned}\tilde{x}^c_{N|k+1} &= A\tilde{x}^c_{N-1|k+1} + B\tilde{u}^c_{N-1|k+1} = \\ &= A\left(\tilde{x}^*_{N|k} + A^{N-1}_K\tilde{w}(x_k)\right) \\ &\quad + B\left(Kx^*_{N|k} + L\tilde{\theta}^*_k + 0 + KA^{N-1}_K\tilde{w}(x_k)\right) = \\ &= A_K\left(\tilde{x}^*_{N|k} + A^{N-1}_K\tilde{w}(x_k)\right) + BL\tilde{\theta}^*_k = \\ &= A_K\tilde{x}^*_{N|k} + BL\tilde{\theta}^*_k + A^N_K\tilde{w}(x_k).\end{aligned}$$

Recall that the terminal set satisfies $\Omega^a_t \subseteq \left\{(x,\theta) : x \in \overline{\mathbb{X}}_N,\ Kx + L\theta \in \overline{\mathbb{U}}_N\right\}$ and

$$\begin{bmatrix} A_K & BL \\ 0 & I \end{bmatrix}\Omega^a_t \oplus \begin{bmatrix} A^N_K \\ 0 \end{bmatrix}\widetilde{\mathbb{W}}(x_k) \subseteq \Omega^a_t.$$

Since $(\tilde{x}^*_{N|k}, \tilde{\theta}^*_k) \in \Omega^a_t$, and

$$\begin{bmatrix} \tilde{x}^c_{N|k+1} \\ \tilde{\theta}^c_{k+1} \end{bmatrix} = \begin{bmatrix} A_K & BL \\ 0 & I \end{bmatrix}\begin{bmatrix} \tilde{x}^*_{N|k} \\ \tilde{\theta}^*_k \end{bmatrix} + \begin{bmatrix} A^N_K \\ 0 \end{bmatrix}\tilde{w}(x_k),$$

it follows that the successor state must remain in the terminal set, i.e., $(\tilde{x}^c_{N|k+1}, \tilde{\theta}^c_{k+1}) \in \Omega^a_t$. Therefore, the candidate solution is necessarily feasible for all constraints in the OCP (20). Thus, $x_k \in \overline{\mathbb{X}}_N \Rightarrow x_{k+1} \in \overline{\mathbb{X}}_N$ for all admissible disturbances $\tilde{w}_k \in \widetilde{\mathbb{W}}(x_k)$, which completes the proof of (i).

2) Proof of Stability and Convergence: By the dual mode prediction paradigm and the construction of the terminal invariant set (23), it follows that the state-dependent uncertainty in mode 2 (i.e., after the states have entered the terminal set) does not have to be accounted for. Therefore, it suffices to prove stability by considering $\tilde{w}(x_{j|k}) = 0$. To

this end, we can follow the method in [29] Appendix A.2 with minor modifications.

The cost function (19) can be related to the infinite horizon nominal cost as

$$V_N(\mathbf{c_k}, \tilde{\theta}_k, x_k, x_k^t) = \tilde{V}(\mathbf{c}_k, x_k) + V_o(\tilde{x}_k^s, x_k^t),$$

where

$$\tilde{V}(\mathbf{c}_k, x_k) = \sum_{j=0}^{\infty} ||x_{j|k} - \tilde{x}_k^s||_Q^2 + ||\tilde{u}_{j|k} - \tilde{u}_k^s||_R^2.$$

The infinite horizon nominal cost, in turn, can be related to the cost of control perturbations $\sum_{j=0}^{N-1} ||c_{j|k}||_\Psi^2$ as follows

$$\begin{aligned}\tilde{V}(\mathbf{c}_k, x_k) &= \sum_{j=0}^{\infty} ||x_{j|k} - \tilde{x}_k^s||_Q^2 + ||\tilde{u}_{j|k} - \tilde{u}_k^s||_R^2 \\ &= \sum_{j=0}^{N-1} ||c_{j|k}||_\Psi^2 + ||x_k - \tilde{x}_k^s||_P^2,\end{aligned}$$

where $\Psi = R + B^\top P B$ and P solves the discrete-time Lyapunov equation $P - A_K^\top P A_K = Q + K^\top R K$ [29], [47]. Since the last term is a constant, minimizing the sum of the control perturbations is equivalent to minimizing the infinite-horizon nominal cost whenever $\Psi = R + B^\top P B$.

To demonstrate stability and convergence, we define the optimal cost function

$$J_k := V_N^*(x_k, x_k^t) = \sum_{j=0}^{N-1} ||c_{j|k}^*||_\Psi^2 + V_o(N_\theta \tilde{\theta}_k^* - x_k^t).$$

Then, we can write

$$\begin{aligned}V_N^c(x_{k+1}, x_{k+1}^t) &= \sum_{j=0}^{N-1} ||c_{j|k+1}^c||_\Psi^2 + V_o(N_\theta \tilde{\theta}_{k+1}^c - x_k^t) \\ &= \sum_{j=1}^{N-1} ||c_{j|k}^*||_\Psi^2 + V_o(N_\theta \tilde{\theta}_k^* - x_k^t) \\ &= J_k - ||c_{0|k}^*||_\Psi^2.\end{aligned}$$

Since this is merely a candidate solution, the optimal solution will yield a cost that is less or equal than $V_N^c(x_{k+1}, x_{k+1}^t)$, i.e., $J_{k+1} \leq V_N^c(x_{k+1}, x_{k+1}^t)$. Therefore,

$$J_{k+1} - J_k \leq -||c_{0|k}^*||_\Psi^2.$$

Thus, $\{J_k\}_{k\geq 0}$ is a non-negative, montonically non-increasing sequence, which must converge to $J_\infty < \infty$ as $k \to \infty$. Summing the difference above from 0 to infinity, we obtain

$$0 \leq \sum_{k=0}^{\infty} ||c_{0|k}^*||_\Psi^2 \leq J_0 - J_\infty < \infty \Rightarrow \lim_{k\to\infty} c_{0|k}^{*\top} \Psi c_{0|k}^* = 0.$$

Since $\Psi > 0$, this implies that $\lim_{k\to\infty} c_{0|k}^* = 0$, which proves (ii).

Define the optimal steady states and inputs as $(\tilde{x}_k^{s,*}, \tilde{u}_k^{s,*}) = M_\theta \tilde{\theta}_k^*$. Then, $u_k = K(x_k - \tilde{x}_k^{s,*}) + \tilde{u}_k^{s,*} + c_{0|k}^*$. We also define deviation variables as $\bar{x} = x_k - \tilde{x}_k^{s,*}$ and $\bar{u} = u_k - \tilde{u}_k^{s,*}$. Substituting into the dynamics equation (2), we obtain

$$\begin{aligned}\bar{x}_{k+1} &= A\bar{x}_k + B\bar{u}_k + \tilde{w}(x_k) \\ &= A_K \bar{x}_k + B c_{0|k}^* + \tilde{w}(x_k).\end{aligned}$$

In addition, we can use the superposition principle as follows

$$\begin{aligned}&\lim_{k\to\infty} \bar{x}_k \\ &= \lim_{k\to\infty} \left[A_K^k \bar{x}_0 + \sum_{j=1}^{k} A_K^{j-1} B c_{0|k-j}^* + \sum_{j=1}^{k} A_K^{j-1} \tilde{w}(x_{k-j}) \right] \\ &= \lim_{k\to\infty} \left[\sum_{j=1}^{k} A_K^{j-1} \tilde{w}(x_{k-j}) \right].\end{aligned}$$

Then, from the definition of the reachable sets, we have

$$\begin{aligned}\lim_{k\to\infty} x_k &\in \lim_{k\to\infty} \left\{\tilde{x}_k^{s,*}\right\} \oplus \mathcal{R}_\infty \\ \lim_{k\to\infty} u_k &\in \lim_{k\to\infty} \left\{\tilde{u}_k^{s,*}\right\} \oplus K\mathcal{R}_\infty\end{aligned}$$

The control actions $c_{j|k} = 0,\ \forall j \in \mathbb{N}_{[0,N-1]}$ are feasible as $k \to \infty$ for any feasible artificial target, since u_k converges to the desired control law inside the terminal set.

As shown above, the optimal cost function J_k consists of two parts: the aggregate contribution from the control actions $c_{j|k}$ and the offset cost $V_o(\cdot)$. Since $\lim_{k\to\infty} c_k^* = 0$ is the global optimum minimizer for the first part of the cost function, it follows that $\lim_{k\to\infty} \tilde{\theta}_k^* = \tilde{\theta}_{\min}$, where $\tilde{\theta}_{\min}$ minimizes the offset cost $V_o(N_\theta \tilde{\theta} - x_\infty^t)$. That is,

$$\tilde{\theta}_{\min} = \arg\min_{\tilde{\theta}} V_o(N_\theta \tilde{\theta} - x_\infty^t).$$

Consequently, this proves (iv). Then, (iii) follows directly from (iv), since whenever the target is reachable the offset cost can be driven to zero by choosing $\bar{x}_s = x_\infty^t$.

C. Online Training

Initially, the structure of the training data is as follows

$$\mathbf{x}_k = \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ \vdots & \vdots \\ x_{k-1,1} & x_{k-1,2} \end{bmatrix}, \quad \mathbf{y}_k = \begin{bmatrix} y_{21}^{\text{mis}} & y_{22}^{\text{mis}} \\ y_{31}^{\text{mis}} & y_{32}^{\text{mis}} \\ \vdots & \vdots \\ y_{k,1}^{\text{mis}} & y_{k,2}^{\text{mis}} \end{bmatrix}.$$

After data from time step $k+1$ becomes available, that is introduced accordingly into $\mathbf{x}$ and $\mathbf{y}$, and the oldest data point is removed in order to keep the size of the datasets constant. Therefore, the new datasets become

$$\mathbf{x}_{k+1} = \begin{bmatrix} x_{21} & x_{22} \\ x_{31} & x_{32} \\ \vdots & \vdots \\ x_{k,1} & x_{k,2} \end{bmatrix}, \quad \mathbf{y}_{k+1} = \begin{bmatrix} y_{31}^{\text{mis}} & y_{32}^{\text{mis}} \\ y_{41}^{\text{mis}} & y_{42}^{\text{mis}} \\ \vdots & \vdots \\ y_{k+1,1}^{\text{mis}} & y_{k+1,2}^{\text{mis}} \end{bmatrix},$$

and the GP regression is re-trained before solving the OCP for the next time step.