Nonlinear model predictive control with explicit back-offs for Gaussian process state space models

Eric Bradford[1], Lars Imsland[2], and Ehecatl Antonio del Rio-Chanona[3]

Abstract— **Nonlinear model predictive control (NMPC) is an efficient control approach for multivariate nonlinear dynamic systems with process constraints. NMPC does however require a plant model to be available. A powerful tool to identify such a model is given by Gaussian process (GP) regression. Due to data sparsity this model may have considerable uncertainty though, which can lead to worse control performance and constraint violations. A major advantage of GPs in this context is its probabilistic nature, which allows to account for plant-model mismatch. In this paper we propose to sample possible plant models according to the GP and calculate explicit back-offs for constraint tightening using closed-loop simulations offline. These then in turn guarantee satisfaction of chance constraints online despite the uncertainty present. Important advantages of the proposed method over existing approaches include the cheap online computational time and the consideration of closed-loop behaviour to prevent open-loop growth of uncertainties. In addition we show how the method can account for updating the GP plant model using available online measurements. The proposed algorithm is illustrated on a batch reactor case study.**

I. INTRODUCTION

Model predictive control (MPC) is an advanced control method that has found a wide range of applications in industry. The success of MPC can be largely attributed to its ability to deal with multivariate plants and process constraints [1]. Linear MPC theory is relatively mature and well-established in practice. Many systems however display strong nonlinear behaviour motivating the use of nonlinear MPC (NMPC) [2]. NMPC is being progressively more utilized due to the advent of improved non-convex optimization algorithms [3], for example in chemical engineering [4].

An important requirement for NMPC is the availability of an accurate plant model. The development of an adequate model has been cited to take up to 80% of the MPC commissioning effort [5]. NMPC algorithms exploit numerous different models, commonly developed by first principles [6]. These are however often too complex and in addition frequently accompanied by high development costs. Alternatively, black-box dynamic models can be used instead, such as support vector machines [7], neural network models [8], or Gaussian processes (GP) [9].

GPs are an interpolation technique developed by [10] that were popularized by the machine learning community [11]. GP predictions take the form of Gaussian distributions. The mean of this distribution can be interpreted as a deterministic prediction, while the variance of the distribution can be seen as a corresponding measure of uncertainty. In particular, an appropriate measure of uncertainty is difficult to obtain by nonlinear parametric models [9]. In control this uncertainty measure can be exploited for efficiently learning a dynamic model by exploring unknown regions or obtaining robustness by avoiding regions that have an uncertainty that is too high. GPs have found various applications in control, such as reinforcement learning [12], designing probabilistic robust linear controllers [13], or for adaptive control [14]. In particular, GPs have been applied in NMPC to notable success as approximate plant models.

The use of GPs for NMPC was first proposed in [15], in which a GP model is updated online for reference tracking without constraints. In [9] the GP is instead identified offline and used online. The variance is constrained to prevent the NMPC from steering the dynamic system into regions with high uncertainty. In [16] GPs are updated online to overcome unmodelled periodic errors, while in [17] the GP is used to update the dynamic model online after a fault has occurred. GPs have also been shown as a useful tool to approximate the mean and variance required in stochastic NMPC [18]. While generally these and other works show the feasibility of GP-based NMPC, there is a lack of efficient methods to account for this uncertainty. Model uncertainty can lead to worse performance and feasibility issues of MPC algorithms, which has led to the development of robust MPC [19] and stochastic MPC [20] approaches.

Most works on GP-based NMPC do consider this uncertainty, however the vast majority of proposed algorithms use stochastic uncertainty propagation to accomplish this, see for example [9], [21], [22]. A recent overview of different stochastic uncertainty propagation approaches can be found in [23]. These approaches have some considerable disadvantages. Firstly, there are no known methods to exactly propagate stochastic uncertainties through the GP model, such that only approximate methods exist usually based on linearization or moment-matching. Secondly, the computational time is often increased significantly due to the stochastic propagation itself. Lastly, most works consider open-loop propagation of uncertainties, which can be prohibitively conservative due to open-loop growth of uncertainties.

Recently there have been some works that do consider other robust control methods. In [24] a robust GP-based NMPC algorithm is developed for learning by propagating ellipsoidal sets using linearization, that provides closed-loop stability guarantees. This approach may however suffer from increased computational times, since the ellipsoidal sets

1 E. Bradford and 2 L. Imsland are with the Department of Engineering Cybernetics, Norwegian University of Science and Technology, Trondheim, Norway, {eric.bradford,lars.imsland}@ntnu.no
3 E.A. del Rio-Chanona is with the Centre for Process Systems Engineering (CPSE), Department of Chemical Engineering, Imperial College London, UK, a.del-rio-chanona@imperial.ac.uk

need to be propagated online. In [25] an alternative procedure is proposed, which establishes closed-loop stability by bounding the one-step ahead error, however determining the required parameters seems to be non-trivial. Lastly, in [26] a robust control approach is proposed for linear systems, in which the GP is used to represent unmodelled non-linearities. The method robustly stabilizes the linear system despite the unmodelled non-linearities, which may however not have a solution if these uncertainties are too large in magnitude.

In this paper we propose a new approach to account for the uncertainty of GP state space models for NMPC for finite-horizon control problems. The method exploits recent results using so-called explicit back-offs, which can be used to account for stochastic uncertainties to design the NMPC [27], [28]. These rely on generating Monte Carlo (MC) closed-loop simulations of possible plant models. The back-offs are then used to tighten the constraints of the NMPC to obtain probabilistic constraint satisfaction despite the stochastic uncertainties present. Further, to generate the required MC samples of the GPs we employ results from [29], [30], where it is shown how to obtain exact samples of GPs. There are several advantages of the proposed method. Firstly, the back-offs are determined using closed-loop simulations, such that the problem of open-loop uncertainty growth is avoided. Further, the required computations are carried-out offline, such that the online computational time is nearly unaffected. In addition, the back-offs are designed based on the empirical cumulative distribution function (cdf), which considers the true underlying distribution [31]. Lastly, due to the independence of the samples some probabilistic guarantees can be given. An extended journal paper of this approach can be found in [32].

The paper is structured as follows. In section II the problem to be solved is outlined. In section III we give an overview of GPs for our purposes. Section IV then outlines the solution approach using GPs. Section V describes the semi-batch reactor case study, for which the results and discussions are given in section VI. Subsequently, section VII concludes the paper.

II. PROBLEM DEFINITION

In this paper we consider a nonlinear discrete-time system:

$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t, \mathbf{u}_t) \tag{1}$$

where t denotes the discrete time, $\mathbf{x} \in \mathbb{R}^{n_x}$ denotes the states, $\mathbf{u} \in \mathbb{R}^{n_u}$ represents the control inputs, and $\mathbf{f} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_x}$ denotes the corresponding nonlinear dynamics.

It is assumed that the full state is measurable with a noisy output measurement of the next state given by:

$$\mathbf{y} = \mathbf{f}(\mathbf{x}, \mathbf{u}) + \boldsymbol{\nu} \tag{2}$$

where $\boldsymbol{\nu} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\boldsymbol{\nu}})$ is independent Gaussian distributed measurement noise with zero mean and a corresponding covariance function $\boldsymbol{\Sigma}_{\boldsymbol{\nu}} = \mathrm{diag}(\sigma^2_{\nu_1}, \ldots, \sigma^2_{\nu_{n_\mathbf{x}}})$.

Let $\mathbf{z} = (\mathbf{x}, \mathbf{u})$ for convenience with joint dimension $n_\mathbf{z} = n_\mathbf{x} + n_\mathbf{u}$. We assume we are given N noisy function evaluations of $\mathbf{f}(\mathbf{z})$ in Eq.(1) according to Eq.(2) denoted as $\mathbf{Y}$ with corresponding input data $\mathbf{Z}$:

$$\mathbf{Z} := [\mathbf{z}_1, \ldots, \mathbf{z}_N]^\mathsf{T} \in \mathbb{R}^{N \times (n_x + n_u)} \tag{3}$$

$$\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_N]^\mathsf{T} \in \mathbb{R}^{N \times n_x} \tag{4}$$

We aim to minimize a finite-horizon cost function:

$$V_T(\mathbf{x}_0, \mathbf{U}) = \sum_{t=0}^{T-1} \ell(\mathbf{x}_t, \mathbf{u}_t) + \ell_f(\mathbf{x}_T) \tag{5}$$

where $T \in \mathcal{N}$ is the time horizon, $\mathbf{U} = [\mathbf{u}_0, \ldots, \mathbf{u}_{T-1}]^\mathsf{T}$ is a collection of control inputs, $\ell : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}$ is the stage cost, and $\ell_f : \mathbb{R}^{n_x} \rightarrow \mathbb{R}$ is the terminal cost.

We assume that the control inputs are subject to hard-constraints, while the states are subject to a joint chance constraint, which can be stated as:

$$\mathbf{u}_t \in \mathbb{U}_t \qquad \forall t \in \{0, \ldots, T-1\} \tag{6}$$

$$\mathbb{P}\left\{ \bigcap_{t=0}^{T} \{\mathbf{x}_t \in \mathbb{X}_t\} \right\} \geq 1 - \epsilon \tag{7}$$

where $\mathbb{X}_t$ are defined as nonlinear constraint sets $\mathbb{X}_t = \{\mathbf{x} \in \mathbb{R}^{n_x} \mid g_j^{(t)}(\mathbf{x}) \leq 0, j = 1, \ldots, n_g\}$

The joint chance constraints are formulated such that the joint event of all $\mathbf{x}_t$ fulfilling the nonlinear constraint sets $\mathbb{X}_t$ is greater than $1 - \epsilon$. It should be noted that the uncertainty in this problem arises from the fact that we do not know $\mathbf{f}(\mathbf{x}, \mathbf{u})$ and are instead given noisy observations of $\mathbf{f}(\mathbf{x}, \mathbf{u})$. We aim to solve this OCP by utilizing GPs to model the unknown dynamics and use the GP approximation to obtain probabilistic guarantees for the closed-loop system.

III. GAUSSIAN PROCESSES

A. Gaussian process regression

In this section we give an introduction to GP regression. For a more general overview refer to [11]. GP regression aims to describe an unknown function $f : \mathbb{R}^{n_z} \rightarrow \mathbb{R}$ using noisy observations y:

$$y = f(\mathbf{z}) + \nu \tag{8}$$

where $\mathbf{z} \in \mathbb{R}^{n_z}$ is the argument of $f()$ and $\nu \sim \mathcal{N}(0, \sigma^2_\nu)$ is Gaussian distributed measurement noise with zero mean and variance σ^2_ν.

GPs consider a distribution over functions, and they can be seen as a generalization of multivariate Gaussian distributions, which can be expressed as:

$$f(\cdot) \sim \mathcal{GP}(m(\cdot), k(\cdot, \cdot)) \tag{9}$$

where the mean function $m(\cdot)$ can be interpreted as the "average" shape of the function, while the covariance function $k(\cdot, \cdot)$ accounts for correlations between function values.

The prior GP is defined by the choice of mean function and covariance function. In this study we apply a zero mean function and the squared-exponential (SE) covariance function [11] [1]:

$$m(\mathbf{z}) := 0 \tag{10}$$

$$k(\mathbf{z},\mathbf{z}') := \alpha^2 \exp\left(-\frac{1}{2}(\mathbf{z}-\mathbf{z}')^\mathsf{T}\mathbf{\Lambda}^{-2}(\mathbf{z}-\mathbf{z}')\right) \tag{11}$$

where $\mathbf{z},\mathbf{z}' \in \mathbb{R}^{n_z}$ are arbitrary input vectors, α^2 is the covariance magnitude, and $\mathbf{\Lambda}^{-2} := \mathrm{diag}(\lambda_1^{-2},\dots,\lambda_{n_z}^{-2})$ is a scaling matrix.

Maximum likelihood estimation is commonly applied to infer the unknown hyperparameters $\mathbf{\Psi} := [\alpha,\lambda_1,\dots,\lambda_{n_z},\sigma_\nu]^\mathsf{T}$, including σ_ν in case the measurement noise variance is also unknown. Consider N noisy function evaluations, denoted by $\mathbf{Y} := [y_1,\dots,y_N]^\mathsf{T} \in \mathbb{R}^N$, with corresponding inputs collected in the matrix $\mathbf{Z} := [\mathbf{z}_1,\dots,\mathbf{z}_N] \in \mathbb{R}^{n_z \times N}$. The log-likelihood of the observed data, ignoring constant terms, is given by:

$$\mathcal{L}(\mathbf{\Psi}) := -\frac{1}{2}\log(\det(\mathbf{K})) - \frac{1}{2}\mathbf{Y}^\mathsf{T}\mathbf{K}^{-1}\mathbf{Y} \tag{12}$$

with $K_{ij} := k(\mathbf{z}_i,\mathbf{z}_j) + \sigma_\nu^2\delta_{ij}$ for each pair $(i,j) \in \{1,\dots,N\}^2$ and the Kronecker delta function δ_{ij}.

The posterior distribution of $f(\mathbf{z})$ at an arbitrary input $\mathbf{z}$, given the input-output data $(\mathbf{Z},\mathbf{Y})$ and the maximum-likelihood estimates of $\mathbf{\Psi}$, follows the Gaussian distribution:

$$f(\mathbf{z})|\mathbf{Z},\mathbf{Y} \sim \mathcal{N}(\mu_f(\mathbf{z};\mathbf{Z},\mathbf{Y}),\sigma_f^2(\mathbf{z};\mathbf{Z},\mathbf{Y})) \tag{13}$$

with

$$\mu_f(\mathbf{z};\mathbf{Z},\mathbf{Y}) := \mathbf{k}(\mathbf{z})\mathbf{K}^{-1}\mathbf{Y} \tag{14}$$

$$\sigma_f^2(\mathbf{z};\mathbf{Z},\mathbf{Y}) := \alpha^2 - \mathbf{k}(\mathbf{z})\mathbf{K}^{-1}\mathbf{k}(\mathbf{z})^\mathsf{T} \tag{15}$$

and $\mathbf{k}(\mathbf{z}) := [k(\mathbf{z},\mathbf{z}_1)\ \cdots\ k(\mathbf{z},\mathbf{z}_N)]$. The mean function $\mu_f(\mathbf{z};\mathbf{Z},\mathbf{Y})$ in this context is the prediction made by the GP at $\mathbf{z}$, while the variance $\sigma_f^2(\mathbf{z};\mathbf{Z},\mathbf{Y})$ provides a measure of uncertainty around this predictor.

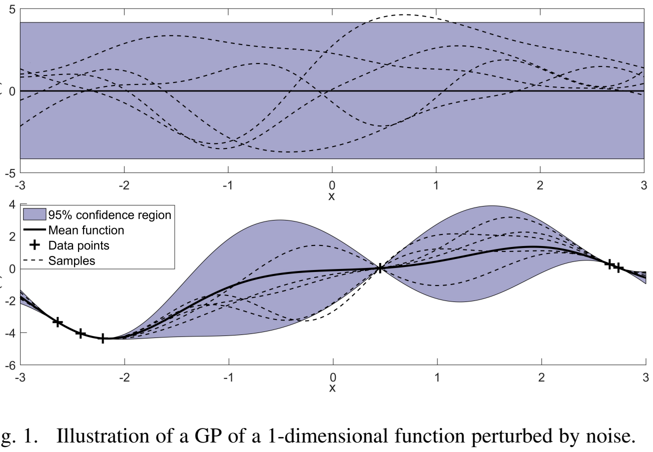

Fig. 1. Illustration of a GP of a 1-dimensional function perturbed by noise. On the top the prior of the GP is shown, while on the bottom the Gaussian process was fitted to several observations to obtain the posterior. The dashed lines show GP samples.

[1]The zero-mean assumption can be achieved by normalizing the data. Also the method presented can be used on any chosen covariance function.

B. Gaussian process state space models

In this section we introduce GP state space models. We aim to identify an unknown state space model from input-output data. GP methodology is usually used to model scalar functions with vector inputs, while for the case of vector functions it is common to build a separate, independent GP for each dimension [12]. Let the function in Eq.(1) be given by $\mathbf{f}(\mathbf{x}_t,\mathbf{u}_t) = [f_1(\mathbf{x}_t,\mathbf{u}_t),\dots,f_{n_x}(\mathbf{x}_t,\mathbf{u}_t)]^\mathsf{T}$, such that we aim to build a separate GP for each function $f_i(\mathbf{x}_t,\mathbf{u}_t)$. We then build a separate GP according to section III-A with observations $\mathbf{Y}_i = [y_{i1},\dots,y_{iN}]^\mathsf{T}$ and inputs $\mathbf{Z}$ for $i \in \{1,\dots,n_x\}$, where y_i refers to the i$^{\text{th}}$ dimension of the measurement $\mathbf{y}$ in Eq.(2). The posterior Gaussian distribution of $\mathbf{f}(\mathbf{x},\mathbf{u})$ given the data-set $(\mathbf{Z},\mathbf{Y})$ in Eq.(3) and Eq.(4) respectively at an arbitrary input $\mathbf{z} = (\mathbf{x},\mathbf{u})$ is given by:

$$\mathbf{f}(\mathbf{z})|\mathbf{Z},\mathbf{Y} \sim \mathcal{N}(\boldsymbol{\mu}_f(\mathbf{z};\mathbf{Z},\mathbf{Y}),\mathbf{\Sigma}_f(\mathbf{z};\mathbf{Z},\mathbf{Y})) \tag{16}$$

with

$$\boldsymbol{\mu}_f(\mathbf{z};\mathbf{Z},\mathbf{Y}) = [\mu_f(\mathbf{z};\mathbf{Z},\mathbf{Y}_1),\dots,\mu_f(\mathbf{z};\mathbf{Z},\mathbf{Y}_{n_x})]^\mathsf{T} \tag{17}$$

$$\mathbf{\Sigma}_f(\mathbf{z};\mathbf{Z},\mathbf{Y}) = \mathrm{diag}\left(\sigma_f^2(\mathbf{z};\mathbf{Z},\mathbf{Y}_1),\dots,\sigma_f^2(\mathbf{z};\mathbf{Z},\mathbf{Y}_{n_x})\right) \tag{18}$$

C. Gaussian process samples

Each sample or realization of a GP in theory yields a deterministic function, however this would require sampling an infinite dimensional stochastic process and hence there are no known methods to obtain an exact sample from a GP. Instead approximate sampling methods have been used, see for example spectral sampling [33]. Exact samples of GPs can however be obtained in the case that the function needs to be evaluated at only a finite number of points, which is commonly the case for state space models. The exact sampling approach was first proposed in [29] and has been applied in [30] for the optimal design of linear controllers.

Assume we are given a GP state space model as in section III-B built from an input data-set $\mathbf{Z}$ and corresponding observations $\mathbf{Y}$. We then wish to create a single GP sample from an initial state $\mathbf{x}_0$ for a finite-horizon T given a known control input sequence $\mathbf{U} = [\mathbf{u}_0,\dots,\mathbf{u}_{T-1}]$. A finite dimensional GP state space model sample over time-horizon T is then represented by a sequence of states. The posterior Gaussian distribution of $\mathbf{x}_1$ is given according to Eq.(16) as:

$$\mathbf{x}_1 = \mathbf{f}(\mathbf{z}_0) \sim \mathcal{N}(\boldsymbol{\mu}_f(\mathbf{z}_0;\mathbf{Z},\mathbf{Y}),\mathbf{\Sigma}_f(\mathbf{z}_0;\mathbf{Z},\mathbf{Y})) \tag{19}$$

Now to obtain a sample of $\mathbf{x}_1$, we draw from the Gaussian distribution in Eq.(19). Let $\mathbf{x}_1^{(s)}$ refer to this realization of $\mathbf{x}_1$, which is considered a realization of the sampled function. To obtain $\mathbf{x}_2^{(s)}$ we then need to first condition on this realization $\mathbf{x}_1^{(s)}$, since it is part of the sampled function path. The realization $\mathbf{x}_1^{(s)}$ is treated similar to a new training point, however without observation noise (i.e. no σ_ν^2 is added to the kernel evaluation $k(\mathbf{z}_0,\mathbf{z}_0)$) and without updating the hyperparameters. If the sampled function revisited the same input, it would lead to the exact same outcome due to the conditioning on a noiseless output.

Now given this new point $\mathbf{x}_1^{(s)}$, we next draw $\mathbf{x}_2^{(s)}$ according to the GP obtained from the updated data-sets. This procedure is recursively repeated until the required time-horizon T has been reached. The overall sampling method is summarized in Algorithm 1 below and is illustrated in Fig.2. This gives us a single GP sample and hence needs to be repeated multiple times to obtain multiple samples.

Algorithm 1: Gaussian process trajectory sampling

```
Input   : $\mathbf{z}_0^{(s)} = \mathbf{z}_0$, $\mathbf{U}$, $\mathbf{K}$, $\mathbf{K}^{-1}$, $\mathbf{Z}$, $\mathbf{Y}$, $\mathbf{k}(\mathbf{z})$ $T$
for each sampling time $t = 1, 2, \ldots, T$ do
    1) Draw $\mathbf{x}_t^{(s)}$ from
       $\mathcal{N}(\boldsymbol{\mu}_f(\mathbf{z}_{t-1}^{(s)}; \mathbf{Z}, \mathbf{Y}), \boldsymbol{\Sigma}_f(\mathbf{z}_{t-1}^{(s)}; \mathbf{Z}, \mathbf{Y}))$
    2) Update $\mathbf{Z} := [\mathbf{Z}^\mathsf{T}, \mathbf{z}_{t-1}^{\mathsf{T}(s)}]^\mathsf{T}$, $\mathbf{Y} := [\mathbf{Y}^\mathsf{T}, \mathbf{x}_t^{\mathsf{T}(s)}]^\mathsf{T}$
    3) Update $\mathbf{K} = \begin{bmatrix} \mathbf{K} & \mathbf{k}(\mathbf{z}_{t-1}^{(s)}) \\ \mathbf{k}(\mathbf{z}_{t-1}^{(s)}) & k(\mathbf{z}_{t-1}^{(s)}, \mathbf{z}_{t-1}^{(s)}) \end{bmatrix}$
    4) Determine $\mathbf{K}^{-1}$
    5) Define $\mathbf{k}(\mathbf{z}) = [\mathbf{k}(\mathbf{z}), k(\mathbf{z}, \mathbf{z}_{t-1}^{(s)})]$
end
Output  : State sequence $\mathbf{x}_1^{(s)}, \ldots, \mathbf{x}_T^{(s)}$
```

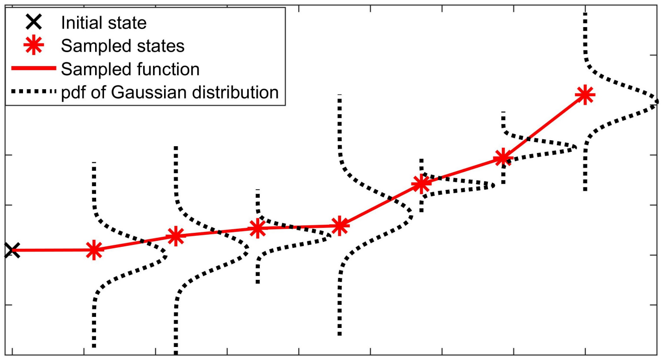

Fig. 2. Illustration of GP sampling scheme for a 1-dimensional function.

IV. SOLUTION APPROACH

Given the input-output data-sets $\mathbf{Z}$ and $\mathbf{Y}$ we fit a GP state space model, see section III-B. We aim to solve the problem defined in section II using NMPC based on this GP model. Now the mean model of the GP defines a state trajectory itself, which we will refer to as the *nominal* case. Each sample of the GP defines further deterministic solutions to the GP state space model. Overall each sample of the GP over a finite horizon T is defined by drawing T independent random states as shown in section III-C, which we will refer to as $\boldsymbol{\Delta}$. For convenience we introduce the notation:

$$\mathbf{x}_t = \boldsymbol{\phi}(t, \mathbf{Y}, \mathbf{Z}; \mathbf{x}, \mathbf{U}, \boldsymbol{\Delta}) \qquad (20)$$

where $\boldsymbol{\phi}(t, \mathbf{Y}, \mathbf{Z}; \mathbf{x}, \mathbf{U}, \boldsymbol{\Delta})$ corresponds to the state at time t, when the initial state is $\mathbf{x}$, the control sequence $\mathbf{U}$ is applied, and the GP realization is given by $\boldsymbol{\Delta}$ given the initial input-output dataset $(\mathbf{Z}, \mathbf{Y})$. Further, by convention let $\boldsymbol{\Delta} = \mathbf{0}$ refer to the *nominal* scenario defined by setting the state space model to the mean function $\boldsymbol{\mu}_f(\mathbf{z}; \mathbf{Y},\mathbf{Z})$.

A. Gaussian process model predictive control

In this section we define the NMPC OCP based on the GP *nominal* model given the data-set $(\mathbf{Z},\mathbf{Y})$. Let the optimization problem be denoted as $P_T(t, \mathbf{Y},\mathbf{Z}; \mathbf{x})$ for the current state $\mathbf{x}$ at discrete-time t:

$$\begin{aligned} &\underset{\mathbf{U}_{t:T-1}}{\text{minimize}} \quad V_T(\mathbf{x}, \mathbf{U}_{t:T-1}) = \sum_{k=t}^{T-1} \ell(\mathbf{x}_k, \mathbf{u}_k) + \ell_f(\mathbf{x}_T) \\ &\text{subject to} \\ &\mathbf{x}_{k+1} = \boldsymbol{\phi}(k, \mathbf{Y}, \mathbf{Z}; \mathbf{x}, \mathbf{U}_{t:k}, \mathbf{0}) \; \forall k \in \{t, \ldots, T-1\} \\ &\mathbf{x}_{k+1} \in \overline{\mathbb{X}}_{k+1}, \quad \mathbf{u}_k \in \mathbb{U}_k \qquad \forall k \in \{t, \ldots, T-1\} \end{aligned} \qquad (21)$$

where $\overline{\mathbb{X}}_{k+1}$ is a tightened constraint set denoted by: $\overline{\mathbb{X}}_{k+1} = \{\mathbf{x} \in \mathbb{R}^{n_x} \mid g_j^{(k)}(\mathbf{x}) + b_j^{(k)} \leq 0\}$, and $\mathbf{U}_{t:k} = [\mathbf{u}_t, \ldots, \mathbf{u}_k]$.

The variables $b_j^{(k)}$ denote so-called back-offs, which aim to tighten the original constraints $\mathbb{X}_t$ to obtain constraint satisfaction for the real plant model using nominal predictions of the state trajectory as in Eq.(21).Let $\mathbf{U}_{t:T}^*(t, \mathbf{Y},\mathbf{Z}; \mathbf{x}) = [\mathbf{u}_t^*(t, \mathbf{Y},\mathbf{Z}; \mathbf{x}), \ldots, \mathbf{u}_{T-1}^*(t, \mathbf{Y},\mathbf{Z}; \mathbf{x})]$ be the optimal control sequence by solving $P_T(t, \mathbf{Y},\mathbf{Z}; \mathbf{x})$. Only the first of these control actions is applied to the plant at time t. This defines our implicit model predictive control law as $\boldsymbol{\kappa}(t, \mathbf{Y},\mathbf{Z}; \mathbf{x}) = \mathbf{u}_t^*(t, \mathbf{Y},\mathbf{Z}; \mathbf{x})$, where the OCP in Eq.(21) needs to be re-solved for each new measurement of $\mathbf{x}_t$.

Applying this closed-loop control policy to a GP sample then leads to the following closed-loop response:

$$\mathbf{x}_t^{\text{MPC}}(\boldsymbol{\Delta}, \mathbf{Y}, \mathbf{Z}) = \boldsymbol{\phi}(t, \mathbf{Y}, \mathbf{Z}; \mathbf{x}_0, \mathcal{K}, \boldsymbol{\Delta}) \qquad (22)$$

where $\mathcal{K} = [\boldsymbol{\kappa}(0, \mathbf{Y}, \mathbf{Z}; \mathbf{x}_0), \ldots, \boldsymbol{\kappa}(t-1, \mathbf{Y}, \mathbf{Z}; \mathbf{x}_{t-1}^{\text{MPC}})]$ is a collection of control actions from the NMPC controller based on observations from the GP plant model given by the realization $\boldsymbol{\Delta}$.

At each time t we are however given a new measurement of $\mathbf{x}_t$ from Eq.(2) and we know the previous input, since it is given by $\mathbf{z}_{t-1} = (\mathbf{x}_{t-1}, \mathbf{u}_{t-1})$. Therefore, it may be reasonable to update the mean/nominal GP model of the MPC using this data online. This leads to the following alternative GP NMPC closed-loop response based on the updated models:

$$\mathbf{x}_t^{\text{MPC,l}}(\boldsymbol{\Delta}, \mathbf{Y}, \mathbf{Z}) = \boldsymbol{\phi}(t, \mathbf{Y}, \mathbf{Z}; \mathbf{x}_0, \mathcal{K}^l, \boldsymbol{\Delta}) \qquad (23)$$

where the collection of control actions from the GP learning NMPC controller is given as $\mathcal{K}^l = [\boldsymbol{\kappa}(0, \mathbf{Y}, \mathbf{Z}; \mathbf{x}_0), \ldots, \boldsymbol{\kappa}(t-1, \mathbf{Y}^{(t-1)}, \mathbf{Z}^{(t-1)}; \mathbf{x}_{t-1}^{\text{MPC,l}})]$ based on observations from the GP plant model given by GP realization $\boldsymbol{\Delta}$. The datasets $\mathbf{Z}^{(t)}$ and $\mathbf{Y}^{(t)}$ are recursively defined as:

$$\mathbf{Z}^{(t)} = [\mathbf{Z}^{\mathsf{T}(t-1)}, \mathbf{z}_{t-1}^\mathsf{T}]^\mathsf{T} \qquad \forall t \in \{1, \ldots, T\} \qquad (24)$$

$$\mathbf{Y}^{(t)} = [\mathbf{Y}^{\mathsf{T}(t-1)}, \mathbf{y}_t^\mathsf{T}]]^\mathsf{T} \qquad \forall t \in \{1, \ldots, T\} \qquad (25)$$

where $\mathbf{y}_t$ is a measurement obtained from Eq.(2).

Note the sampling procedure of the plant model is unchanged, but the learning GP NMPC algorithm is updated based on the most recent measurements. This leads to the mean function being updated in a similar procedure to the GP sampling in Algorithm 1, however in this case the noise σ_ν^2 is included in the kernel evaluation. The hyperparameters are kept at their nominal value due to the excessive computational cost required to update these. The closed-loop trajectories defined by Eq.(22) and Eq.(23) are tied to the choice of the tightened constraint set $\overline{\mathbb{X}}$.

B. Back-off constraints

In this section we outline how to determine the back-off constraints in Eq.(21) to obtain probabilistic constraint satisfaction of the real plant as defined in the problem definition in section II based on the GP description of the plant. The GP provides both a nominal model, but also describes a distribution of many possible plant models based on the initial data-set given. The overall aim is to determine back-off constraints and corresponding tightened constraint sets $\overline{\mathbb{X}}_t$, such that the closed-loop response given by either Eq.(22) for GP NMPC without learning or Eq.(23) for GP NMPC with learning satisfies the constraint set $\mathbb{X}_t$ with a high probability. Note that the learning GP NMPC has a different closed-loop behaviour from the GP NMPC without learning, such that these yield different back-off values.

We propose to use S independent samples of the GP generated using the procedure in section III-C, which then in turn describe S different possible plant models. The closed-loop response of these is then given by either the GP NMPC without learning:

$$\mathbf{x}_t^{\text{MPC}}(\mathbf{\Delta}^{(s)}) = \boldsymbol{\phi}(t; \mathbf{x}_0, \mathcal{K}, \mathbf{\Delta}^{(s)}) \quad \forall s \in \{1, \dots, S\} \quad (26)$$

or the GP NMPC with learning:

$$\mathbf{x}_t^{\text{MPC,l}}(\mathbf{\Delta}^{(s)}) = \boldsymbol{\phi}(t; \mathbf{x}_0, \mathcal{K}^l, \mathbf{\Delta}^{(s)}) \quad \forall s \in \{1, \dots, S\} \quad (27)$$

The aim is now to ensure that $\mathbf{x}_t^{\text{MPC}}(\mathbf{\Delta}^{(s)})$ satisfies $\mathbb{X}_t$ for all t, for all but a few samples to attain a high probability of constraint satisfaction.

It is however very difficult to derive an update rule for the back-off constraints on joint probabilities, i.e. based on the intersection of $\mathbb{X}_k$. Instead, we propose an update rule that is applied point-wise to each nonlinear constraint $g_j^{(t)}(\mathbf{x}_k^{\text{MPC}}(\mathbf{\Delta}^{(s)}))$ following a procedure proposed in [28]. Assume we aim to determine back-off constraints that imply:

$$g_j^{(t)}(\mathbf{x}_k^{\text{MPC}}(\mathbf{0})) + b_j^{(t)} = 0 \implies \mathbb{P}\{g_j^{(t)}(\mathbf{x}_k^{\text{MPC}}(\mathbf{\Delta})) \leq 0\} \geq \delta \quad (28)$$

i.e. the back-offs are adjusted such that the satisfaction of the constraints given the nominal model predictions implies satisfaction of the other possible plant models according to the GP distribution with a probability of at least δ.

We then define the empirical cumulative distribution function (ecdf) as:

$$\hat{F}_{g_j^{(t)}} = \frac{1}{S}\sum_{i=1}^{S} \mathbf{1}\{g_j^{(t)}(\mathbf{x}_t^{\text{MPC}}(\mathbf{\Delta}^{(s)})) \leq 0\} \quad (29)$$

where $\hat{F}_{g_j}^{(t)}$ is a sample approximation of the chance constraint given in Eq.(28) on the RHS.

In [28] it is proposed to iteratively update the back-offs based on Eq.(28) using the inverse of the ecdf in Eq.(29) [2]. We then iterate over n_b back-off iterations using the approach given in algorithm 2.

Algorithm 2: Back-off iterative updates

```
for $n_b$ back-off iteration do
    for each sampling time $t = 1, 2, \dots, T$ and
      constraints $j = 1, \dots, n_g^{(t)}$ do
        Initialize: Set $b_j^{(t)}$ to some reasonable values
          1) Run $S$ simulations of either Eq.(26) without
             learning or Eq.(27) with learning
          2) Update $b_j^{(t)} := \hat{F}^{-1}_{g_j^{(t)}}(\delta) - g_j^{(t)}(\mathbf{x}_k^{MPC}(\mathbf{0}))$
    end
end
Output : $b_j^{(t)}$
```

C. Probabilistic guarantees

In this section we give some probabilistic guarantees for the problem definition in section II given fixed back-off values. It should be noted that our problem is posed as a finite-horizon control problem, such that the NMPC implementation has a shrinking horizon. Given S independent GP samples, we can define the following ecdf to approximate the joint probability in Eq.(7):

$$\hat{F}_{\mathbb{X}} = \frac{1}{S}\sum_{i=1}^{S} \mathbf{1}\left\{\bigcap_{t=0}^{T}\{\mathbf{x}_t \in \mathbb{X}_t\}\right\} \quad (30)$$

where $\hat{F}_{\mathbb{X}}$ is essentially equal to the fraction of trajectories, which violate the joint constraint given in Eq.(7).

Theorem 1 below, which was derived in [31] can be used to obtain probabilistic guarantees based on the ecdf of independent samples. It is therefore important that the sampling of the GP is carried-out using independent MC samples as shown in section III-C.

Theorem 1. *If $\beta_{cor} = \hat{F}_{\mathbb{X}}$ based on S independent MC samples, then the following lower bound holds on the true probability $\beta = \mathbb{P}\left\{\bigcap_{t=0}^{T}\{\mathbf{x}_t \in \mathbb{X}_t\}\right\}$ with a probability of at least $1-\alpha$:*

$$\beta_{lb} = 1 - \textit{betainv}\left(\alpha, S + 1 - \lfloor \beta_{cor} S \rfloor, \lfloor \beta_{cor} S \rfloor\right) \geq \beta \quad (31)$$

The operator $\lfloor\rfloor$ denotes rounding towards $-\infty$, and betainv denotes the inverse of the cumulative Beta-distribution.

Feasibility of the original chance constraint in Eq.(1) then follows trivially from Theorem 1 as stated below.

Corollary 1. *For an unknown plant model that follows the GP distribution identified exactly, such that the uncertainty description of the GP is accurate, and given a $\beta_{lb} \geq 1 - \epsilon$ with a probability of $1-\alpha$ that fulfills Theorem 1, then the chance constraint in Eq.(1) holds with a probability no smaller than $1-\alpha$.*

[2]The inverse of ecdf is given by the quantile function.

D. Algorithm

```
Algorithm 3: Back-off GP NMPC
  Offline Computations
    1) Build GP state-space model from data-sets $\mathbf{Z}, \mathbf{Y}$.
    2) Choose initial condition $\mathbf{x}_0$, stage costs $\ell$ and $\ell_f$
       and constraint sets $\mathbb{X}_t, \mathbb{U}_t \; \forall t \in \{1, \ldots, T\}$
    3) Determine explicit back-off constraints using
       algorithm 2 with or without learning.
    4) Check probabilistic guarantees as shown in section
       IV-C to obtain $\epsilon$.
  Online Computations
  for $t = 0, \ldots, T$ do
      1) Solve the MPC problem in Eq.(21)
      2) Apply the first control input of the optimal solution
      3) Measure the state $\mathbf{x}_t$ and update
         the GP plant model for learning GP NMPC.
  end
```

V. CASE STUDY

In this paper we apply the approach to a challenging semi-batch reactor case study adopted from [34], which is an important example of a finite-horizon control problem in chemical engineering. The following series chemical reactions take place in the reactor catalyzed by H_2SO_4:

$$2\text{A} \xrightarrow[(1)]{k_{1\text{A}}} \text{B} \xrightarrow[(2)]{k_{2\text{B}}} 3\text{C}$$

The reactions taking place are all first-order. Chemical reaction (1) is an exothermic reaction, while chemical reaction (2) is endothermic. A cooling jacket is used for temperature control. The control variables are given by the flowrate of pure reactant A entering the reactor and the temperature of the cooling jacket T_0. Overall there are 5 states: concentrations of reactants A, B, and C in mol/L, reactor temperature in K, and the reactor volume in L.

The objective of the case study is to maximize the amount of product C at the final time horizon. In addition, there are two path constraints. Firstly, the reactor temperature needs to be kept below 420K for safety reasons and the volume needs to stay below 800L, which is the capacity of the reactor. The evolution of these states can be described by a differential algebraic equation system, which can be found in [34]. The time horizon T is fixed to 10 with a sampling time of 0.4h giving an overall batch run-time of 4h. The concentrations of A, B and C is initially zero with an initial reactor temperature of 290K and a volume of 100L.

VI. RESULTS AND DISCUSSIONS

In this section the results based on the case study outlined in section V are presented. Overall 6 different scenarios were run for verification of the approach. We fit an initial GP model using data-sets with 50, 100, and 150 datapoints (dtps) according to a space-filling Latin hypercube design. Further, each of these was applied to the algorithm presented in section IV-D with online learning and without online learning. The explicit back-offs were determined using 400 GP MC samples for each back-off iteration. The back-offs were adjusted in a total of 5 iterations using algorithm 2. Lastly, for the final back-off values the GP NMPC was applied to the real plant represented by the case study equations. The results of these simulations are summarized in Figs.3-7 and in Table I.

In Fig.3 and Fig.4 plots are shown of the closed-loop volume and temperature trajectories according to the 400 MC samples of the plant model. The top two graphs of each figure show the trajectories for 50 data points, while the bottom two graphs show it for 100 data points. On the left the graphs show the trajectories considering online learning, while on the right without online learning. We can see that the back-offs are adjusted such that most trajectories are kept below the constraint shown by the red line. Nonetheless, for the trajectories using only 50 data-points many trajectories overshoot the temperature constraint by a significant amount due to the inability of our method to find consistently good back-off values due to the very high uncertainty. For 150 data-points on the other hand the frequency of constraint violation is relatively low. In addition, it can be seen that the learning based method is able to more quickly reach the temperature constraint and also stay closer to it leading to improved performance and less conservative back-offs.

In Fig.5 and Fig.6 the closed-loop response of the GP NMPC is shown for the ”real” plant using the exact case study equations. In the top two graphs of each figure the response is shown using back-offs, while in the bottom figures the response is displayed disregarding back-offs. Further, the left figures show the trajectories without learning, while the right figures utilize the available data. Firstly, it can be seen that disregarding back-offs leads to constraint violations of volume or temperature for all cases due to the mismatch between the ”real” plant and the GP approximation. This in turn is avoided in all cases employing back-offs, except for 50 data points, which overshoots the temperature constraint by a significant amount. This is somewhat expected, since as shown in Fig.4 the determined back-offs are inadequate. Lastly, it can be seen that in all 3 cases using back-offs the trajectories using online learning are able to stay closer to the temperature constraint.

Apart from constraint satisfaction, it is also important to look at the performance of the different approaches in terms of the economic objective achieved. This is shown in Fig.7, where box-plots are given for the 6 different cases from the objective values obtained at the final back-off iterations. The aim of the NMPC is to maxmize the amount of product C. In general one would assume that more data leads to an improved objective value, since then the GP model used is closer to the real plant. We can see that this is mostly true apart for the 50 data points run, which however as mentioned has inadequate back-offs and therefore overshoots the constraints by a large margin. In addition we can see that for both 100 data points and 150 data points learning performs on average better than not learning, which is as

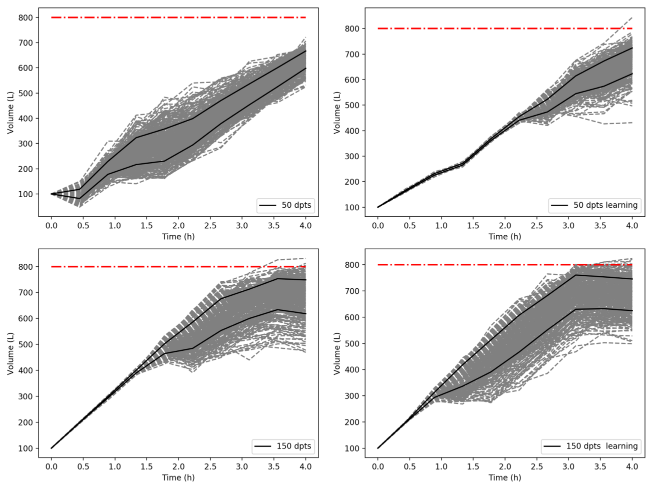

Fig. 3. Closed-loop trajectories of volume for 400 GP MC samples for the cases 50 and 150 data-points with and without learning.

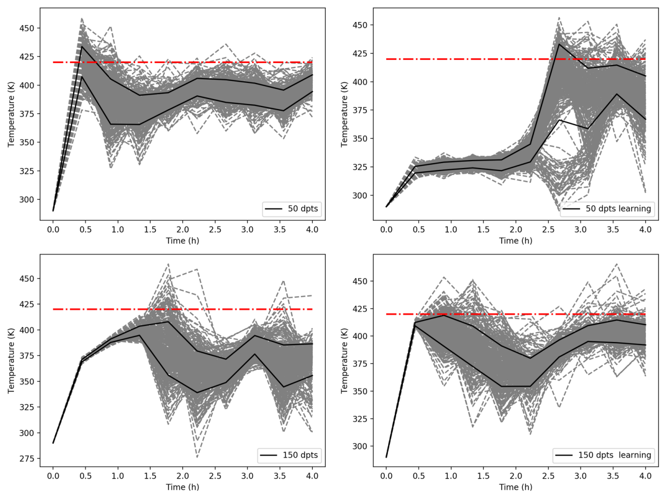

Fig. 4. Closed-loop trajectories of temperature for 400 GP MC samples for the cases 50 data-points and 150 data-points with and without learning.

expected since it is less conservative and should have a plant model closer to the real plant model.

Lastly, in Table I we show the probability of constraint violation stated as ”Probability”, which corresponds to the fraction of the 400 GP NMPC MC trajectories that violated either temperature or volume constraint for the final back-off iteration. ”c-Probability” refers to the real guaranteed probability βlb of violation using the theorem outlined in section IV-C. We can see a clear trend that the more data we have, the smaller the probability of constraint violation, which is more or less as expected. In particular, for 50 data points without learning we have more MC samples violating the constraints than not, while for 150 data points we are able to guarantee constraint satisfaction of nearly 0.9. In addition, we see that for 50, 100, and 150 data points learning can provide significantly higher probability guarantees than their counter-parts without learning. In addition, computationally times are on-average low in the range of seconds, however they do rapidly increase with the number of data-points.

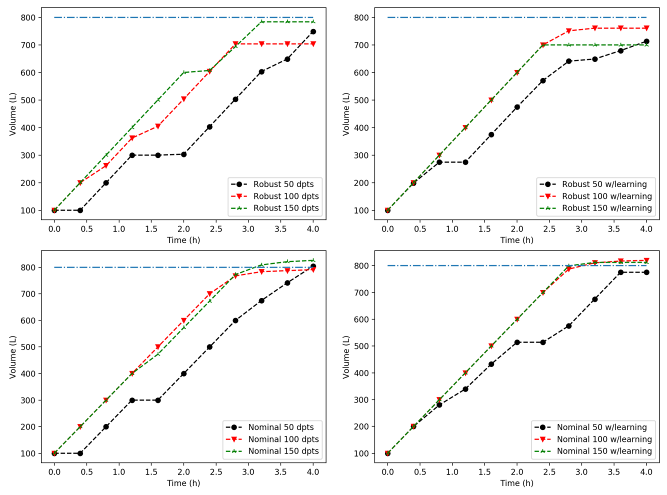

Fig. 5. Closed-loop trajectories of volume for the ”real” plant with tightened constraints at the top and without tightened constraints at the bottom.

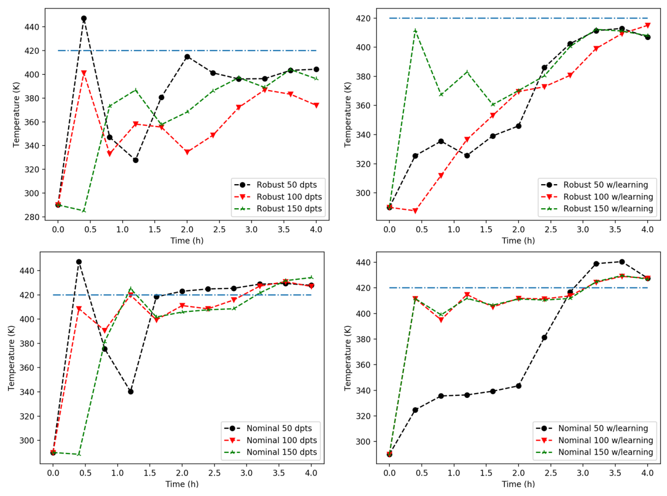

Fig. 6. Closed-loop trajectories of temperature for the ”real” plant with tightened constraints at the top and without tightened constraints at the bottom.

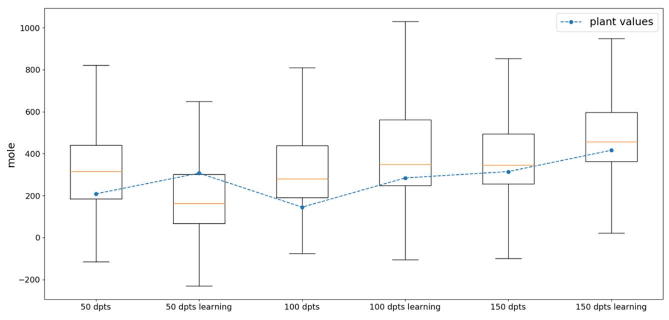

Fig. 7. Box plot of objective values from 400 MC closed-loop simulations of the final back-off iteration together with the obtained objectives for the ”real” plant shown in blue.

TABLE I

COMPARISON OF CLOSED-LOOP CONSTRAINT SATISFACTION,CORRECTED GUARANTEED PROBABILITY, AND AVERAGE OCP/NMPC SOLUTION TIME.

Set-up	**Probability**	**c-Probability**	**OCP time (s)**
50 dpts	0.55	0.57	0.11
50 dpts learning	0.31	0.33	0.19
100 dpts	0.27	0.30	0.52
100 dpts learning	0.21	0.24	0.75
150 dpts	0.16	0.19	1.12
150 dpts learning	0.09	0.11	1.34

VII. CONCLUSIONS

In conclusion, a new approach has been proposed for finite-horizon control problems using NMPC in conjunction with GP state space models. The method utilizes the probabilistic nature of GPs to sample deterministic functions of possible plant models. Tightened constraints using explicit back-offs are then determined, such that the closed-loop simulations of these possible plant models is feasible to a high probability. In addition it is shown how probabilistic guarantees can be derived based on the number of constraint violations from the simulations. It was in addition shown that online learning can be accounted for explicitly in this method, which leads to overall less conservativeness. Lastly, the computational times could be shown to be relatively low, since the constraint tightening is performed offline.

ACKNOWLEDGMENTS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 675215.

REFERENCES

[1] J. M. Maciejowski, *Predictive control: with constraints*. Pearson education, 2002.

[2] F. Allgöwer, R. Findeisen, and Z. K. Nagy, "Nonlinear model predictive control: From theory to application," *J. Chin. Inst. Chem. Engrs*, vol. 35, no. 3, pp. 299–315, 2004.

[3] L. T. Biegler, *Nonlinear programming: concepts, algorithms, and applications to chemical processes*. Siam, 2010, vol. 10.

[4] L. T. Biegler and J. B. Rawlings, "Optimization approaches to nonlinear model predictive control," Tech. Rep., 1991.

[5] Z. Sun, S. J. Qin, A. Singhal, and L. Megan, "Performance monitoring of model-predictive controllers via model residual assessment," *Journal of Process Control*, vol. 23, no. 4, pp. 473–482, 2013.

[6] Z. K. Nagy, B. Mahn, R. Franke, and F. Allgöwer, "Real-time implementation of nonlinear model predictive control of batch processes in an industrial framework," in *Assessment and Future Directions of Nonlinear Model Predictive Control*. Springer, 2007, pp. 465–472.

[7] X.-C. Xi, A.-N. Poo, and S.-K. Chou, "Support vector regression model predictive control on a HVAC plant," *Control Engineering Practice*, vol. 15, no. 8, pp. 897–908, 2007.

[8] S. Piche, B. Sayyar-Rodsari, D. Johnson, and M. Gerules, "Nonlinear model predictive control using neural networks," *IEEE Control Systems Magazine*, vol. 20, no. 3, pp. 53–62, 2000.

[9] J. Kocijan, R. Murray-Smith, C. E. Rasmussen, and A. Girard, "Gaussian process model based predictive control," in *American Control Conference, 2004. Proceedings of the 2004*, vol. 3. IEEE, 2004, pp. 2214–2219.

[10] D. G. Krige, "A statistical approach to some basic mine valuation problems on the Witwatersrand," *Journal of the Southern African Institute of Mining and Metallurgy*, vol. 52, no. 6, pp. 119–139, 1951.

[11] C. E. Rasmussen, "Gaussian processes for machine learning," 2006.

[12] M. Deisenroth and C. E. Rasmussen, "PILCO: A model-based and data-efficient approach to policy search," in *Proceedings of the 28th International Conference on machine learning (ICML-11)*, 2011, pp. 465–472.

[13] F. Berkenkamp and A. P. Schoellig, "Safe and robust learning control with Gaussian processes," in *2015 European Control Conference (ECC)*. IEEE, 2015, pp. 2496–2501.

[14] G. Chowdhary, H. A. Kingravi, J. P. How, and P. A. Vela, "Bayesian nonparametric adaptive control using gaussian processes," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 3, pp. 537–550, 2015.

[15] R. Murray-Smith, D. Sbarbaro, C. E. Rasmussen, and A. Girard, "Adaptive, cautious, predictive control with Gaussian process priors," *IFAC Proceedings Volumes*, vol. 36, no. 16, pp. 1155–1160, 2003.

[16] E. D. Klenske, M. N. Zeilinger, B. Schölkopf, and P. Hennig, "Gaussian process-based predictive control for periodic error correction," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 1, pp. 110–121, 2016.

[17] J. M. Maciejowski and X. Yang, "Fault tolerant control using Gaussian processes and model predictive control," in *Control and Fault-Tolerant Systems (SysTol), 2013 Conference on*. IEEE, 2013, pp. 1–12.

[18] E. Bradford and L. Imsland, "Stochastic Nonlinear Model Predictive Control Using Gaussian Processes," in *2018 European Control Conference (ECC)*. IEEE, 2018, pp. 1027–1034.

[19] A. Bemporad and M. Morari, "Robust model predictive control: A survey," in *Robustness in identification and control*. Springer, 1999, pp. 207–226.

[20] A. Mesbah, "Stochastic model predictive control: An overview and perspectives for future research," *IEEE Control Systems*, vol. 36, no. 6, pp. 30–44, 2016.

[21] L. Hewing, A. Liniger, and M. N. Zeilinger, "Cautious NMPC with Gaussian Process Dynamics for Autonomous Miniature Race Cars," in *2018 European Control Conference (ECC)*, 2018, pp. 1341–1348.

[22] G. Cao, E. M.-K. Lai, and F. Alam, "Gaussian process model predictive control of an unmanned quadrotor," *Journal of Intelligent & Robotic Systems*, vol. 88, no. 1, pp. 147–162, 2017.

[23] L. Hewing and M. N. Zeilinger, "Cautious Model Predictive Control using Gaussian Process Regression," *arXiv preprint arXiv:1705.10702*, 2017.

[24] T. Koller, F. Berkenkamp, M. Turchetta, and A. Krause, "Learning-based model predictive control for safe exploration and reinforcement learning," *arXiv preprint arXiv:1803.08287*, 2018.

[25] M. Maiworm, D. Limon, J. M. Manzano, and R. Findeisen, "Stability of gaussian process learning based output feedback model predictive control," *IFAC-PapersOnLine*, vol. 51, no. 20, pp. 455–461, 2018.

[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] R. W. Koller, L. A. RicardezSandoval, and L. T. Biegler, "Stochastic backoff algorithm for simultaneous design, control, and scheduling of multiproduct systems under uncertainty," *AIChE Journal*, vol. 64, no. 7, pp. 2379–2389, 2018.

[28] J. A. Paulson and A. Mesbah, "Nonlinear model predictive control with explicit backoffs for stochastic systems under arbitrary uncertainty," *IFAC-PapersOnLine*, vol. 51, no. 20, pp. 523–534, 2018.

[29] S. Conti, J. P. Gosling, J. E. Oakley, and A. O'Hagan, "Gaussian process emulation of dynamic computer codes," *Biometrika*, vol. 96, no. 3, pp. 663–676, 2009.

[30] J. Umlauft, T. Beckers, and S. Hirche, "Scenario-based Optimal Control for Gaussian Process State Space Models," in *2018 European Control Conference (ECC)*. IEEE, 2018, pp. 1386–1392.

[31] S. Streif, M. Karl, and A. Mesbah, "Stochastic nonlinear model predictive control with efficient sample approximation of chance constraints," *arXiv preprint arXiv:1410.4535*, 2014.

[32] E. Bradford, L. Imsland, D. Zhang, and E. A. d. R. Chanona, "Stochastic data-driven model predictive control using Gaussian processes," *arXiv preprint arXiv:1908.01786*, 2019.

[33] E. Bradford, A. M. Schweidtmann, and A. Lapkin, "Efficient multiobjective optimization employing Gaussian processes, spectral sampling and a genetic algorithm," *Journal of Global Optimization*, vol. 71, no. 2, pp. 407–438, 2018.

[34] E. Bradford and L. Imsland, "Economic Stochastic Model Predictive Control Using the Unscented Kalman Filter," *IFAC-PapersOnLine*, vol. 51, no. 18, pp. 417–422, 2018.