On the use of supervised clustering in stochastic NMPC design

Mazen Alamir

Technical Notes and Correspondence

On the Use of Supervised Clustering in Stochastic NMPC Design

Mazen Alamir

***Abstract*—In this article, a supervised clustering-based heuristic is proposed for the real-time implementation of approximate solutions to stochastic nonlinear model predictive control frameworks. The key idea is to update online a low cardinality set of uncertainty vectors to be used in the expression of the stochastic cost and constraints. These vectors are the centers of uncertainty clusters that are built using the optimal control sequences, cost, and constraints indicators as supervision labels. The use of a moving clustering data buffer which accumulates recent past computations enables to reduce the computational burden per sampling period while making available at each period a relevant amount of samples for the clustering task. A relevant example is given to illustrate the contribution and the associated algorithms.**

***Index Terms*—.**

I. Introduction

Stochastic nonlinear model predictive control (SNMPC) is without doubt one of the major challenges facing the NMPC community for the years to come. This can be viewed as the third *key step* to achieve. Indeed, after the 90s where the provable stability was the main paradigm [9], the last ten years or so were dedicated to making available reliable, and easy to use NMPC solvers for nominal deterministic settings [4]. The success of these two steps helped propelling MPC-based solutions *out-of-labs* toward the real-life paradigm where the keywords are *risk*, *uncertainties,* and *probability*.

After some early attempts involving robust NMPC [8] which rapidly appeared to be over stringent, it quickly becomes obvious that the natural way to address the new paradigm is to replace all the MPC ingredients (cost, constraints) by their *expected* counterparts in the formulation of the open-loop optimization problem. Stochastic NMPC was born for which excellent recent unifying reviews can be found in [10], [12], and [13].

Unfortunately, the apparently intuitive and simple shift in paradigm consisting in doing the *business as usual* on the expected quantities, comes with heavy consequences in terms of computational burden. Indeed, computing the expectation of a nonlinear function of several variables for each candidate control sequence is obviously an impossible task. Only approximations can be attempted, each coming with its own merits and drawbacks.

The first idealistic option is to use the stochastic dynamic programming (SDP) framework which is based on the well known Bellman's principle of optimality in which the conditional probability plays the role of extended state [13]. Unfortunately, solving the SDP leads to algorithms that scale exponentially in the dimension of the state. Nevertheless, for small sized problems, nice and elegant solutions can be derived [16] that might even address realistic real-life problems.

A second option is to derive online a structured approximation (Gaussian processes or chaos polynomials for instance) of the probability density function at the current state and then to use the resulting approximation in evaluating the expectation of relevant quantities [6] and [14]. Note however that this has to be done for all possible candidate control sequences in each iteration of the NLP solver. This obviously restricts the field of application of this approach to small-sized and rather slow systems if any.

The third and probably more pragmatic option is to use scenarios-based averaging in order to approximate the expectations (or optionally higher order moments) involved in the problem formulation [17]. In this case, a high number (say K) of samples of the random quantities is drawn and the resulting constraints and state equations are concatenated while sharing the same control. A common *optimal* control sequence is then searched for using standard nominal solvers.

This last approach may lead to a very high dimensional problem that is not intuitively prone to a parallel computing or distribution over the system life-time. This is especially true when the underlying (deterministic) problem is solved using efficient multiple-shooting algorithms [5] since the dimension of the extended state is proportional to the number of samples K being involved. The latter can be quite high in order to get a decent level of certification [3]. Moreover, the need to introduce variance-related terms in the formulation to better address the chance constraint certification [11] makes things even worse as double summation on the set of scenarios has to be performed leading to a K^2-rated complexity.

It is worth underlstanding that even when putting aside the computational challenges associated to SNMPC, one has to keep in mind that all these methods assume that the statistical description of the uncertainty vector is available (to draw relevant samples) and that the problem lies in the way to propagate it depending on the control actions. This knowledge is never available and can only be presumed. This should achieve convincing us that we need to accept a painful transition from a proof-related certain paradigm to a realm of heuristics which can only be evaluated once implemented and its results diagnosed on real-life problems. Consequently, the implementability/scalability issues become crucial and key properties of any solution framework to SNMPC.

The present article addresses the scenario-based SNMPC framework under this last point of view, namely, that of implementable and scalable heuristics.

An overview of the framework proposed in the present contribution is sketched in Fig. 1. In this figure, x, u, w, J, and g refer to state, control, uncertainty, cost function, and constraint, respectively. The basic block (at the bottom of Fig. 1) where SNMPC is performed is the grayed box

Manuscript received November 7, 2019; accepted January 22, 2020. Date of publication; date of current version. This work was supported by MIAI @ Grenoble Alpes under Grant ANR-19-P3IA-0003. Recommended by Associate Editor Prof. L. Zhang.

The author is with the University Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, 38000 Grenoble, France (e-mail: mazen.alamir@grenoble-inp.fr).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAC.2020.2970424

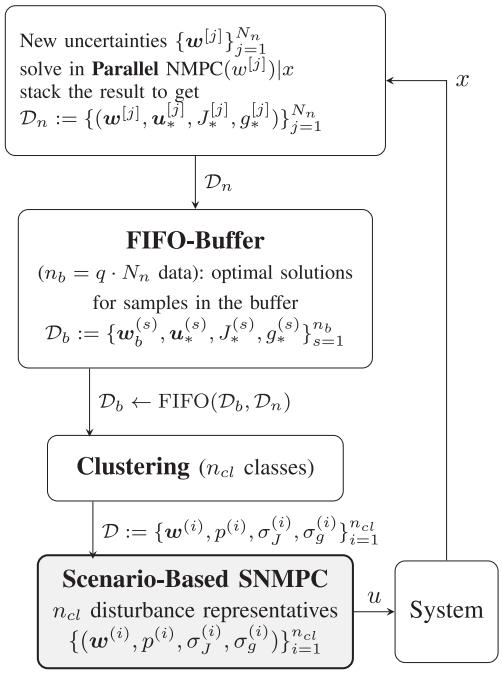

Fig. 1. Schematic view of the proposed SNMPC framework.

that delivers the action to be applied to the controlled system, namely, the first action in the scenario-based optimal sequence.

The key difference with the standard implementation is that the SNMPC is formulated using only a few number (n_{cl}) of regularly updated disturbance samples. More precisely, n_{cl} is the number of clusters used in a clustering step. This clustering box delivers to the SNMPC box a regularly updated set containing the centers of clusters together with their population weights ($p^{(i)}$) and dispersion indicators ($\sigma_J^{(i)}, \sigma_g^{(i)}$) in the data set $\mathcal{D}_b$ used to achieve the clustering task. This data set $\mathcal{D}_b$ is accumulated in a first in first out (FIFO) buffer. The latter receives at each updating step a new block of data $\mathcal{D}_n$ which is delivered by the top block. This data block $\mathcal{D}_n$ contains a set of N_n nominal solutions $\boldsymbol{u}_*^{[j]}$ of a standard NMPC with presumably known newly sampled disturbance vectors $w^{[j]}$ together with the corresponding optimal costs and constraints indicators $J_*^{[j]}, g_*^{[j]}$, $j = 1, \ldots, N_n$. As the N_n optimization problems are totally decoupled, the computation performed in this top block can be done in fully parallel way.

The rational behind this framework lies in the intuition that very often, while the space of possible uncertainty realizations might be very rich (including uniform distributions in high dimensional hypercubes), the set of corresponding optimal ingredients (control sequences, optimal cost, constraints indicators) might accept a low cardinality set of meaningfully distinct clusters. Moreover, the loss of information that results from using only the centers of clusters in the formulation can be partially mitigated by using the statistical information $(\sigma_J^{(i)}, \sigma_g^{(i)})$, $i = 1, \ldots, n_{cl}$ regarding the dispersion of cost and constraints indicator within each cluster. This information is transmitted from the clustering layer as indicated in Fig. 1. Sections III-B and III-C give more detailed description of the above two steps.

The aim of this article is to give a rigorous presentation of this framework and to propose a complete implementation on a relevant example in order to assess the performance and implementability of the framework.

This article is organized as follows. Section II gives some definitions and notation used in the sequel. The proposed framework is explained in Section III by successively explaining the different boxes depicted in Fig. 1. An illustrative example is given in Section IV. Finally, Section V concludes the article and gives some hints for further investigation.

II. Definitions and Notation

We consider nonlinear dynamic systems given by

$$x^+ = f(x, u, w) \tag{1}$$

where $x \in \mathbb{R}^n$, $u \in \mathbb{U} \subset \mathbb{R}^m$, and $w \in \mathbb{R}^r$ stand, respectively, for the vectors of state, control, and uncertainty. It is assumed here for simplicity that the whole state vector is measured while the uncertainty is not. Moreover, it is also assumed that the size of the uncertainty vector and the level of excitation are such that the uncertainty estimation through dedicated observer is not a reasonable option.

Consider that a couple of cost/constraints functions can be defined at any current state x by[1] $J^{(x)}(\boldsymbol{u}, \boldsymbol{w}) \in \mathbb{R}_+$ and $g^{(x)}(\boldsymbol{u}, \boldsymbol{w}) \leq 0 \in \mathbb{R}$ that express, respectively, a cost function to be minimized (in some sense) and a constraints violation indicator to be limited (in some sense) over some finite prediction horizon of length N and starting from the initial state x. When the state is implicitly known (or fixed during some argumentation), the short notation $J(\boldsymbol{u}, \boldsymbol{w})$ and $g(\boldsymbol{u}, \boldsymbol{w})$ can be used.

Remark 2.1: Note that g is a scalar map that might encompass a set of constraints to be enforced through dedicated maps (such as $\max\{0, \cdot\}^2$ for instance). The treatment of this function can be vectorized for computational efficiency (including by using of a vector of slack variables in softening the constraints rather than the scalar used in the sequel). We keep nevertheless this scalar notation for the sake of simplicity of exposition of the main ideas. In the simulation however, vectorized implementation is used.

The ideal stochastic NMPC formulation that is approximated in the present article takes the following form:

$$\min_{(\boldsymbol{u},\mu)} \mathbb{E}(J^{(x)}(\boldsymbol{u}, \cdot)) + \left[\frac{1-\epsilon_J}{\epsilon_J}\right] \mathbb{S}(J^{(x)}(\boldsymbol{u}, \cdot)) + \rho\mu \tag{2}$$

$$\text{under } \mathbb{E}(g^{(x)}(\boldsymbol{u}, \cdot)) + \left[\frac{1-\epsilon_g}{\epsilon_g}\right] \mathbb{S}(g^{(x)}(\boldsymbol{u}, \cdot)) \leq \mu \geq 0 \tag{3}$$

which can be understood by means of the following comments:

✓ $\mathbb{E}$ and $\mathbb{S}$ denote, respectively, the expectation and the standard deviation of their arguments over the presumably known statistics on the uncertainty vector $\boldsymbol{w}$.

✓ According to [11], under some technical conditions,[2] when $\mu = 0$ the satisfaction of (3) implies that the probability of satisfaction of the original constraint $g^{(x)}(\boldsymbol{u}, \boldsymbol{w}) \leq 0$ is greater than $1 - \epsilon_g$ and this, regardless of the specific statistics of the uncertain variables. Using μ with a high penalty ρ implements a soft version of this formulation.

✓ Similarly, under the same conditions, the cost function that is minimized in (2) when using $\rho = 0$, is precisely the bound below which it can be certified, with a probability greater than $1 - \epsilon_J$, that the expectation of the cost lies.

The difficulty in implementing a solution to the formulation (2)–(3) lies in the cost of approximating the expectation and standard deviation involved. The commonly used approaches replace the expectation by an averaging sum over a high number of uncertainties samples which can be quite heavy to compute as mentioned in the introduction. In the following section, the proposed approximating method to the formulation (2)–(3) is described.

[1] Boldfaced notation $\boldsymbol{x}$, $\boldsymbol{u}$, and $\boldsymbol{w}$ are used to denote variables profiles over some prediction horizon.

[2] Although these conditions might not be satisfied, the idea is used here to support the proposed heuristic

III. Proposed Framework

In this section, the different tasks involved in the framework depicted in Fig. 1 are successively detailed.

A. Solving a Set of Deterministic Problems: Construction of a New Data Set $\mathcal{D}_n$

This task consists in drawing a new set of N_n values $\boldsymbol{w}^{[j]}$ of the uncertainty profile and to solve, knowing these values, the corresponding individual deterministic constrained optimization problem given by

$$\boldsymbol{u}_*^{[j]} \leftarrow \min_{\boldsymbol{u},\mu} J^{(x)}(\boldsymbol{u}, \boldsymbol{w}^{[j]}) + \rho\mu \mid g^{(x)}(\boldsymbol{u}, \boldsymbol{w}^{[j]}) \leq \mu \tag{4}$$

the resulting individual optimal cost and constraints are denoted by $J_*^{[j]}$ and $g_*^{[j]}$, respectively. This enables the following data set to be defined:

$$\mathcal{D}_n := \{(\boldsymbol{w}^{[j]}, \boldsymbol{u}_*^{[j]}, J_*^{[j]}, g_*^{[j]})\}_{j=1}^{N_n}. \tag{5}$$

Note that solving these individual problems while knowing the values of the disturbance profiles enables to reveal a *population of control sequences* that would be optimal should the disturbance profiles that originates them occurs. The relevance of this computation is to use the resulting data in a disturbance-profiles clustering step. This is because

> The disturbance profiles that correspond to *similar* optimal control sequences, optimal cost and constraint values, should be declared to lie in the same cluster of disturbance profiles even if they strongly differ as a high dimensional vectors.

Because the clustering is based on the labels constituted by the triplet $(\boldsymbol{u}_*^{[j]}, J_*^{[j]}, g_*^{[j]})$, the clustering is qualified hereafter as a supervised clustering.

Note that this step is totally parallelizable as the individual deterministic problems are totally decoupled. Nevertheless, the number N_n of samples can be moderate since a buffer is created and updated by such data at each sampling period as explained and justified in the next section.

Since the dataset $\mathcal{D}_n$ is related to a current state x_k at instant k, it is denoted by $\mathcal{D}_n(k)$ when the reference to the sampling instant k is needed.

B. Updating the Clustering Buffer: Creating and Updating the Dataset $\mathcal{D}_b$

This is a simple FIFO data storage task in which the successive datasets $\mathcal{D}_n$ of the form (5) are stacked for use in the clustering task.

As the new datasets $\mathcal{D}_n(k)$ are added at each sampling period k, the size $n_b := qN_n$ of the clustering buffer (q is the number of successive datasets $\mathcal{D}_n(k)$ to be included) depends on the bandwidth of the system. This is because integrating all the datasets $\mathcal{D}_n(k)$ in a single clustering dataset (called $\mathcal{D}_b$ in Fig. 1) ignores the fact that each of these datasets is related to a different state that defines the underlying optimization problem (4). The underlying assumption is that the evolution of the state during the q successive sampling periods can be viewed as sufficiently small for the clustering dataset $\mathcal{D}_b$ to remain relevant.

To summarize, at each sampling period $k > q$, the clustering data set is given by

$$\mathcal{D}_b(k) := \{\mathcal{D}_n(k-1), \ldots, \mathcal{D}_n(k-q)\} \tag{6}$$

where $\mathcal{D}_n(k-j)$ is the dataset containing the solutions of the N_n nominal problems defined by (4) with the state x_{k-j}. For smaller initial values of k, the buffer contains only the available $k-1$ datasets $\mathcal{D}_n(k-1), \ldots, \mathcal{D}(0)$.

Remark 3.1: The choice of the size q of the clustering set $\mathcal{D}_b$ is obviously the object of a recurrent type of dilemmas commonly encountered in real-time MPC. This dilemma holds between the quality of the solution of a problem (better if the size of the cluster is large) and the very relevance of the problem itself (weak if the buffer contains data that correspond to too old states). In nominal deterministic MPC, the parameter to be tuned is the number of iterations of the underlying optimization algorithm [2].

Having the clustering data $\mathcal{D}_b$, the next section explains the supervised clustering task that leads to the selection of the n_{cl} clusters whose centers form the database $\mathcal{D}$ feeding the SNMPC formulation (Fig. 1).

C. Clustering the Uncertainty Set: Creating and Updating the Low Cardinality Dataset $\mathcal{D}$

Clustering is a key branch of Data Mining whose objective is to split a set of data into *subsets* such that inside each subset, the data are similar in some sense (according to some distance). Obviously, the clustering topic is vast and it is outside the scope of the present contribution to give a survey of available clustering techniques. Readers can consult [18] for a comprehensive and recent survey.

Fortunately enough, when it comes to use clustering (or more generally many machine learning) algorithms as parts of a wider solution framework (as it is the case in the present contribution), free publicly available implementations of clustering task can be used such as the well-known scikit-learn library [15].

A clustering map C takes as arguments as follows

1) a disctere set $\mathcal{V} := \{v^{(i)}\}_{i=1}^{n_b}$ to be split into clusters;
2) an integer n_{cl} representing the number of clusters which one wishes $\mathcal{V}$ to be split into

and delivers as output a n_b-dimensional vector of *labels* $\mathcal{I} \in \{1, \ldots, n_{cl}\}^{n_b}$ that associates to each member $v^{(i)}$ of $\mathcal{V}$ its associated cluster. This is shortly written as follows:

$$\mathcal{I} = C(\mathcal{V}, n_{cl}) \in \{1, \ldots, n_{cl}\}^{n_b}. \tag{7}$$

Recall that our objective is to perform a clustering of the set of disturbance vectors $\mathcal{W} := \{w_b^{(s)}\}_{s=1}^{n_b}$ contained in the dataset $\mathcal{D}_b$ (see Fig. 1).

Clustering algorithms (K-Means, Mean-shift, DBSCAN, to cite but few algorithms in the scikit-learn library) generally perform an *unsupervised learning* in the sense that they consider only internal relationships and distances between the elements of the set $\mathcal{V}$ to split and this is regardless of any exogenous information[3] about these elements.

Following the discussion of Section III, we seek a clustering that considers as *similar* those disturbance vectors that correspond to *similar* triplets of control profiles, cost, and constraint indicators. This is the reason why the set $\mathcal{V}$ that is used hereafter is given by

$$\mathcal{V} = \{(u_*^{(s)}, J_*^{(s)}, g_*^{(s)})\}_{s=1}^{n_b}. \tag{8}$$

That is why we refer to the proposed clustering approach as a *supervised clustering* as the set of class labels $\mathcal{I}$ that will be used to split the uncertainty vectors set is derived using the *exogenous* information contained in the set $\mathcal{V}$ given by (8), namely

$$\mathcal{I} := C\Big(\{(u_*^{(s)}, J_*^{(s)}, g_*^{(s)})\}_{s=1}^{n_b}, n_{cl}\Big). \tag{9}$$

[3]Called *labels* in the machine learning language.

Once this clustering is achieved, the centers of the n_{cl} resulting clusters are given as a by-side product of the clustering task

$$w^{(i)} := \text{Mean}\left(w_b^{(s)}, s \in \{1,\dots,n_b\} \mid \mathcal{I}_s = i\right). \tag{10}$$

Beside these centers, the weights of the different clusters can be associated to the relative size of their populations, namely

$$p^{(i)} := \frac{1}{n_b}\text{card}\left\{s \in \{1,\dots,n_b\} \mid \mathcal{I}_s = i\right\}. \tag{11}$$

Finally, evaluations of the dispersions of the cost function and the constraints inside each cluster can also be cheaply obtained using the two variances defined by

$$\sigma_J^{(i)} := \text{Var}\left(J_*^{(s)}, s \in \{1,\dots,n_b\} \mid \mathcal{I}_s = i\right) \tag{12}$$

$$\sigma_g^{(i)} := \text{Var}\left(g_*^{(s)}, s \in \{1,\dots,n_b\} \mid \mathcal{I}_s = i\right). \tag{13}$$

This ends the definition of the dataset $\mathcal{D}$ (see Fig. 1) that is used in the formulation of the SNMPC which is described in the following section.

D. Formulation of the Stochastic NMPC

In this section, approximate expressions for (2) and (3) are given using the ingredients contained in the dataset $\mathcal{D}$ which is updated at each sampling period using the steps explained in the previous sections. This is done by averaging over the set of centers $w^{(i)}$, $i = 1,\dots,n_{cl}$ of the clusters created above while accommodating for the dispersion inside the clusters. More precisely, the following optimization problem is considered:

$$\min_{\boldsymbol{u},\mu}\left[\hat{\mathbb{E}}^{(x)}(\boldsymbol{u}) + \frac{1-\epsilon_J}{\epsilon_J}\hat{\mathbb{S}}^{(x)}(\boldsymbol{u}) + \rho\mu\right] \tag{14}$$

under the following constraints:

$$g^{(x)}(\boldsymbol{u}, w^{(i)}) + \frac{1-\epsilon_g}{\epsilon_g}\sqrt{\sigma_g^{(i)}} \le \mu \ge 0 \qquad \forall i \in \{1,\dots,n_{cl}\} \tag{15}$$

in which:

$$\hat{\mathbb{E}}^{(x)}(\boldsymbol{u}) := \sum_{i=1}^{n_{cl}} p^{(i)}\left[J^{(x)}(\boldsymbol{u}, w^{(i)}) + \frac{1-\epsilon_J}{\epsilon_J}\sqrt{\sigma_J^{(i)}}\right] \tag{16}$$

$$\hat{\mathbb{S}}^{(x)}(\boldsymbol{u}) := \left(\sum_{i=1}^{n_{cl}} p^{(i)}\left[J^{(x)}(\boldsymbol{u}, w^{(i)}) + \frac{1-\epsilon_J}{\epsilon_J}\sqrt{\sigma_J^{(i)}} - \hat{\mathbb{E}}^{(x)}(\boldsymbol{u})\right]^2\right)^{\frac{1}{2}} \tag{17}$$

where an estimation $\hat{\mathbb{E}}^{(x)}(\boldsymbol{u})$ of the expectation of the cost function (corresponding to the term $\mathbb{E}(J^{(x)}(\boldsymbol{u},\cdot))$ in (2)) is computed using a $p^{(i)}$-based weighted sums in which the predicted values at the center of the clusters are augmented by the terms that depend on the estimated variances $\sigma_J^{(i)}$ included in the dataset $\mathcal{D}$ as described above. Similarly, the term $\hat{\mathbb{S}}^{(x)}(\boldsymbol{u})$ is used as an approximation of the standard deviation term $\mathbb{S}(J^{(x)}(\boldsymbol{u},\cdot))$ in (2).

As for the expression (15) of the constraints, note that thanks to the low number of clusters n_{cl}, one can afford to enforce the personalized approximated expression of (3) on each cluster individually rather than taking the global statistics over all the clusters. That is the reason why the constraints are enforced on each individual cluster by tightening the bounds by an amount that is proportional to the standard deviation within each cluster as estimated from the precomputed parameters $\sigma_g^{(i)}$ for $i = 1,\dots,n_{cl}$. This leads to a harder tightening of the constraints over those clusters showing higher dispersions.

E. Discussion Regarding the Parameters Choice

The choice of the parameters q, N_n, n_{cl} is guided by the following issues.

1) As already underlined in Remark 3.1, the choice of q is determined by the need to keep the buffer D_b relevant. This is because this buffer contains data that correspond to q past sampling instants τ. Therefore, denoting by $\tau_{\max}$ the maximum time during which one can reasonably consider the state as *almost unchanged*, the following constraint needs to be satisfied:
$$q \le \lfloor \frac{\tau_{\max}}{\tau} \rfloor.$$
2) On the other hand, assuming that the computation of the N_n optimization problems in $\mathcal{D}_n$ is done in parallel and denoted by the following.
   a) $\tau^c_{\text{clustering}}(qN_n)$: the time needed to perform the clustering task (which depends on the number of elements qN_n).
   b) $\tau^c_{\text{MPC}}(n_{cl})$: the time needed to solve the scenario-based MPC involving the n_{cl} centers of the clusters. Note that this time increases mainly because of the concatenation of the constraints over each cluster.
   c) $\tau_u = N_u\tau$: the control updating period, namely the time between two successive resolutions of the scenario-based MPC problem.

   Then the following constraint should be satisfied for the scheme to be feasible:
$$\tau^c_{\text{clustering}}(qN_n) + \tau^c_{\text{MPC}}(n_{cl}) + \tau^c_{\text{MPC}}(1) \le \tau_u. \tag{18}$$
3) Obviously, depending on the problem, the system cannot be left in open-loop more than some time duration $\tau_u^{\max}$, therefore, the following constraint holds on the r.h.s of (18):
$$\tau_u \le \tau_u^{\max}. \tag{19}$$
4) In addition to (18), the scalability of the kernel stochastic MPC problem can be evaluated by keeping in mind that the use of n_{cl} clusters multiplies the number of constraints by n_{cl} while keeping the same number of decision variables when single shooting method is used while it multiply both the constraints and the number of decision variables by n_{cl} in a multiple shooting framework.

IV. Illustrative Example: Combined Therapy of Cancer

A. System Equations, Objective, and Constraints

As an illustrative example, let us consider the problem of drug dosing during a combined chemotherapy/immunotherapy of cancer [1] and [7]. The dynamic model involves 4 states, 2 control inputs, and 13 uncertain parameters. More precisely, the state components are defined as follows:

x_1 tumor cell population;
x_2 circulating lymphocytes population;
x_3 chemotherapy drug concentration;
x_4 effector immune cell population;
u_1 rate of introduction of immunotherapy drug;
u_2 rate of introduction of chemotherapy drug.

and the dynamics is given by

$$\dot{x}_1 = ax_1(1-bx_1) - c_1x_4x_1 - k_3x_3x_1 \tag{20}$$

$$\dot{x}_2 = -\delta x_2 - k_2x_3x_2 + s_2 \tag{21}$$

$$\dot{x}_3 = -\gamma_0 x_3 + u_2 \tag{22}$$

$$\dot{x}_4 = g\frac{x_1}{h+x_1}x_4 - rx_4 - p_0x_4x_1 - k_1x_4x_3 + s_1u_1. \tag{23}$$

TABLE I
NOMINAL VALUES OF THE PARAMETERS

param	value	param	value	param	value
a	$0.25\ day^{-1}$	b	$1.02 \times 10^{-14}\ cell^{-1}$	c_1	$4.41 \times 10^{-10}\ (cell \cdot day)^{-1}$
g	$1.5 \times 10^{-2}\ day^{-1}$	h	$2.02 \times 10^{1}\ cell^{2}$	k_2, k_3	$6 \times 10^{-1}\ day^{-1}$
k_1	$8 \times 10^{-1}\ day^{-1}$	p_0	$2 \times 10^{-11}\ (cell \cdot day)^{-1}$	s_1	$1.2 \times 10^{7}\ cell \cdot day^{-1}$
s_2	$7.5 \times 10^{6}\ cell \cdot day^{-1}$	δ	$1.2 \times 10^{-2}\ day^{-1}$	γ	$9 \times 10^{-1}\ day^{-1}$
r	$4.0 \times 10^{-2}\ cell \cdot day^{-1}$				

The description of the relevance of all the terms and coefficients can be examined in [1] although one can easily guess from the definition of the state components.

In order to understand the necessary tradeoff and timing issues, note that the immunotherapy drug (u_1) enables to increase the effector immune population size x_4 [see (23)] which enhances the tumor decrease through the second term ($-c_1 x_4 x_1$) in (20). On the other hand the chemotherapy drug (u_2) does not only attack the tumor cells through the term ($-k_3 x_3 x_1$) in (20) but also the effector immune population x_4 as well as the circulating lymphocytes x_2 through the terms $-k_1 x_4 x_3$ and $-k_3 x_3 x_1$ respectively present in (23) and (20). These coupled effects render mandatory the computation of a rationale timing and tradeoff that is made fragile in the presence of high uncertainties on all the parameters.

Using the notation above, the uncertainty vector w gathers all the uncertain parameters involved in the model (20)–(23), namely

$$w := \Big[a, b, c_1, k_3, \delta, k_2, s_2, \gamma_0, g, r, p_0, k_1, s_1\Big] \in \mathbb{R}_+^{13} \tag{24}$$

that are supposed here to be constant but unknown. Note also that a reconstruction of all these parameters from patient measurement during the treatment is obviously out of question. Table I gives the nominal values of the parameters involved in the dynamics. Note that because of the excursion of these parameters and the related states, a normalized version of the dynamics (20)–(23) is derived by using the following vector of normalization values of the state components:

$$\bar{x} := \Big[10^9, 10^9, 1, 10^9\Big]. \tag{25}$$

As it is typically the case in cancer treatment, the control objective is to reduce the tumor cells population x_1 at the end of the treatment while ensuring that the health of the patient (represented in the above model by the circulating lymphocytes population size x_2) remains greater than some a priori fixed lower bound $x_2^{\min}$.

Consequently, the following cost function is used at each state x in the MPC design:

$$J^{(x)}(\boldsymbol{u}, w) := \rho_f x_1(N) + \sum_{i=1}^{N} x_1(i|\boldsymbol{u}, x, w) + \rho_u |u(i)| \tag{26}$$

where $x(i|\boldsymbol{u}, x, w)$ is the state i sampling period ahead when starting from the initial state x while applying the control sequence $\boldsymbol{u}$ under the parameter vector w. The above function has to be minimized while enforcing the following constraint on the predicted trajectory:

$$g^{(x)}(\boldsymbol{u}, w) := \min_{i \in \{1,\dots,N\}} [x_2(i|\boldsymbol{u}, x, w)] \geq x_2^{\min}. \tag{27}$$

The control input is saturated according to $u \in [0, 5] \times [0, 1]$.

B. Stochastic MPC Controller Settings

In all the forthcoming simulations, the sampling period $\tau = 0.2$ (Days) is used. When stochastic MPC is used, the number of clusters is taken equal to $n_{cl} = 3$. The number $N_n = 25$ of new samples is generated at each sampling period (see Fig. 1). The size of the FIFO buffer is taken equal to $n_b = 4 \times N_n = 100$ ($q = 4$). The parameters ϵ_J and ϵ_g used to account for the variance in the definition of the cost function and the constraints are taken equal to $\epsilon_J = \epsilon_g = 0.1$ (leading to 90% of confidence rate). The weighting coefficients $\rho_f = 1000$, $\rho_u = 1$, and $\rho = 10$ are used. The clustering is performed using the KMeans module of the scikit-learn python library [15].

The stochastic MPC is compared to the nominal MPC which uses the nominal values of the parameters as given in Table I. As for the stochastic MPC, the random values of these parameters are obtained according to

$$w_i = (1 + \nu_i)\bar{w}_i \quad \text{where } \nu_i \in \mathcal{N}(0, \sigma) \tag{28}$$

where a variance $\sigma = 0.2$ is used leading to samples that might have a discrepancy as high as 45–80% of the nominal values. Hundred simulations are performed using either stochastic or nominal MPC and statistical indicators are compared. Note that the cloud of disturbances used in these 100 simulations are *fired* independently of those fired to feed the FIFO buffer of the stochastic MPC. All the simulations use the normalized initial state $x_0 = (1.0, 0.15, 0, 1)$ and all the simulations last 40 d. The prediction horizon length is taken equal to $N = 10$ (2 d) and five steps of the optimal control sequence is applied before a new optimal sequence is computed. This leads to an updating control period τ_u of 1 d. The problem encoding and the optimization are performed using multiple-shooting formulation (with warm starting of the initial guess at each sampling period) thanks to the free software CasADi [4] (python version) on a MacBookPro 2.9 GHz Intel Core i7.

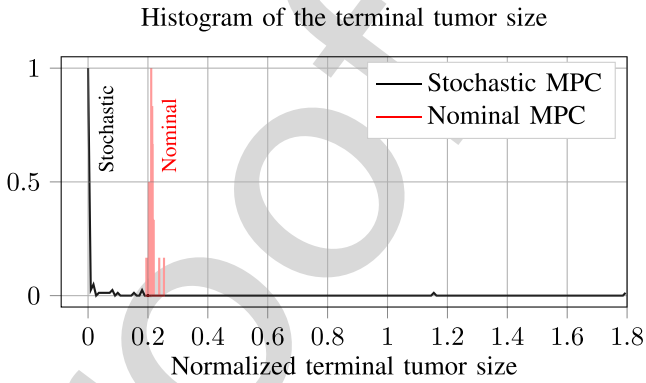

Fig. 2. Histogram of the terminal normalized tumor sizes under nominal and stochastic MPC controllers.

C. Results and Discussion

Fig. 2 shows the normalized (w.r.t the maximum bins) histograms of the tumor sizes at the end of the closed-loop simulations. This figure shows that the SNMPC outperforms the nominal MPC as it leads to a vanishing tumor size except for two single outliers where the tumor is increased as explained later on. Fig. 3 shows the normalized histogram

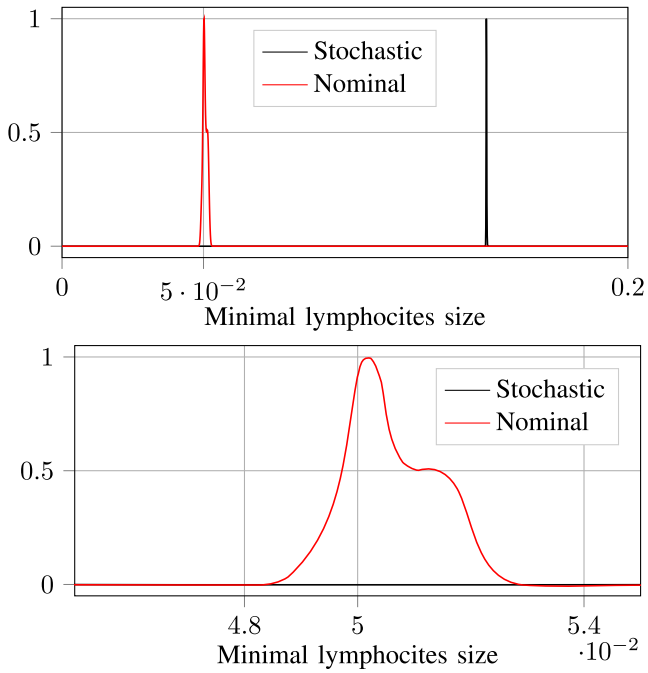

Fig. 3. (top): Histogram of the minimal lymphocytes population size. (Bottom): Zoom on the nominal histogram showing constraints violation in 15% of the scenarios. Note that the bottom plot is a zoom on the top plot around the lower bound $x_2^{\min} = 0.05$.

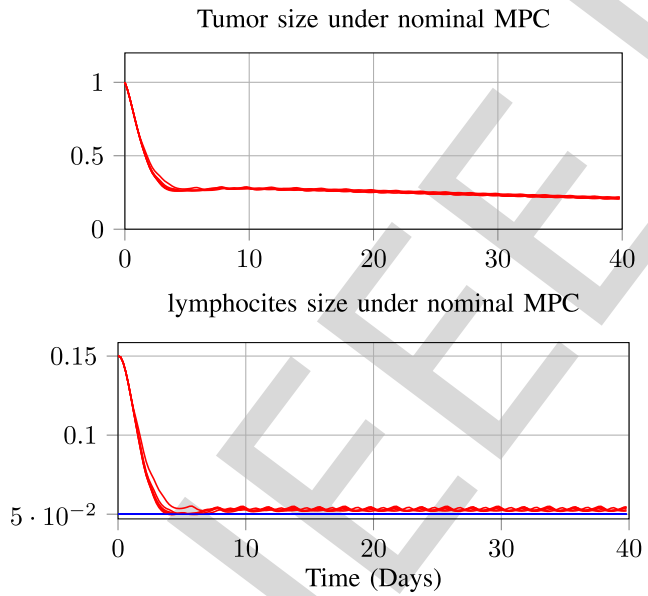

Fig. 4. Typical closed-loop behavior under the nominal MPC.

of the minimal lymphocytes population's size during the closed-loop simulations. Note how the bottom plot of this figures shows that under the nominal MPC, the constraints is violated in around 15% of the scenarios. Note however how the constraints is largely respected when the SNMPC is used due to the cautious behavior of the stochastic controller.

One might notice here that something uncommon is happening as the stochastic controller wins on both sides, namely the cost function and the constraints satisfaction. This can be explained by examining

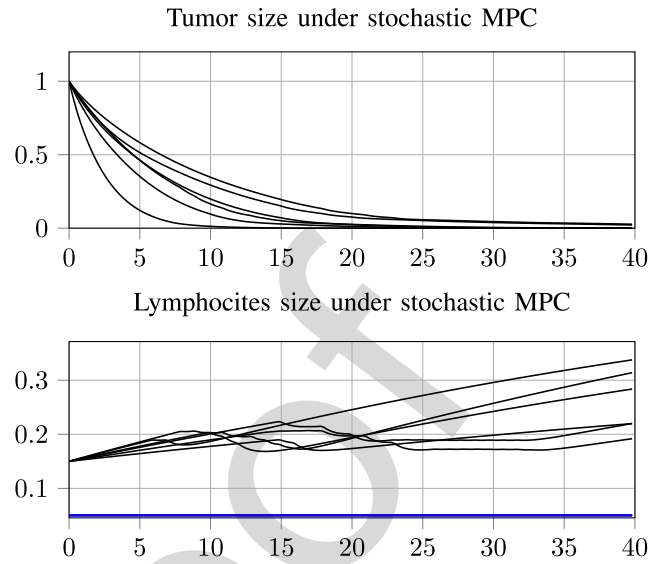

Fig. 5. Typical closed-loop behavior under the stochastic MPC.

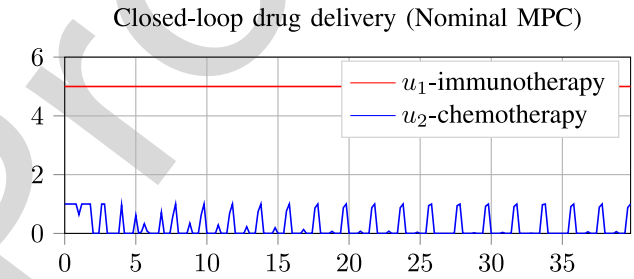

Fig. 6. Typical drug delivery under nominal MPC.

the typical behavior of the closed-loop under the nominal versus the stochastic MPC controllers which are depicted, respectively, in Figs. 4 and 5. As a matter of fact, since the nominal controller is not cautious and does not see the risk of violating the constraints, it applies intensive chemotherapy drug from the beginning as this reduced the tumor size quickly and hence lead to a lower value of the cost function. But when the horizon recedes, the closed-loop system is trapped since there is no more possibility to reduce the cost significantly without violating the constraint on the lymphocytes population size. That is the reason why the nominal controller can only regulate the lymphocytes size by applying in parallel chemotherapy and immunotherapy (see the Fig. 6).

On the other hand, the stochastic MPC does not fall in this *trap* as handling all the clusters representative makes it aware of a high risk of constraints violation in case intensive chemotherapy is used from the beginning. That is the reason why, it applies chemotherapy only after a while when the level of lymphocytes becomes high enough to ensure a secure delivery of chemotherapy drug. This can be clearly seen in Figs. 5 and 7.

Note that a longer prediction horizon could have brought the nominal controller into the same strategy than the stochastic one avoiding thus the above mentioned trap. Indeed, longer prediction horizon would have revealed that better contraction can be achieved while increasing the health indicator should the application of the chemotherapy be delayed. The choice of the scenario is here to illustrate the difference between the two settings and the capabilities of SNMPC to enforce the satisfaction of the constraints when compared to a nominal MPC. Finally, Table II shows the comparison between the statistics of the computation time

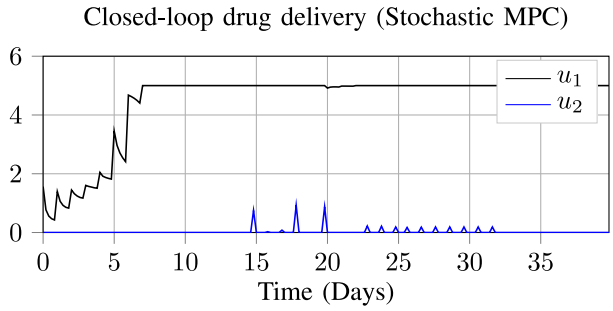

Fig. 7. Typical drug delivery under stochastic MPC.

TABLE II
STATISTICS OF THE COMPUTATION TIMES (IN MS) OF A SINGLE SOLUTION OF THE OPTIMIZATION PROBLEM. (CASADI [4] (PYTHON VERSION) ON A MACBOOKPRO 2.9 GHZ INTEL CORE I7)

	Mean (ms)	**Standard deviation** (ms)
Nominal	270	76
Stochastic	383	330

that is needed to solve the underlying optimization problems at each control updating period. This table clearly shows that using $n_{cl} = 3$ cluster induces on average an extra computational burden of 40% while increasing the dispersion of the computation to a higher extent. This also suggests that without using the clustering approach, the computation time would be too prohibitive.

V. CONCLUSION AND FUTURE WORK

In this article, a clustering-based framework is proposed to derive an approximated version of nonlinear stochastic MPC control design that is illustrated on a realistic and challenging examples involving a high dimensional nonreconstructible uncertainty vector. Work in progress targets a better understanding of the way the number of clusters can be rationally chosen, the impact of the choice of the labels involved in the clustering step, and the size of the FIFO buffer (the forgetting rate of previous samples). Application to many other real-life examples is also under investigation.

REFERENCES

[1] M. Alamir, "On probabilistic certification of combined cancer therapies using strongly uncertain models," *J. Theor. Biol.*, vol. 384, pp. 59–69, 2015.

[2] M. Alamir, "A state-dependent updating period for certified real-time model predictive control," *IEEE Trans. Autom. Control*, vol. 62, no. 5, pp. 2464–2469, May 2017.

[3] T. Alamo, R. Tempo, and E. Camacho, "Randomized strategies for probabilistic solutions of uncertain feasibility and optimization problems," *IEEE Trans. Autom. Control*, vol. 54, no. 11, pp. 2545–2559, Nov. 2009.

[4] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: A software framework for nonlinear optimization and optimal control," in *Mathematical Programming Computation*. New York, NY, USA: Springer, 2018. Q11

[5] H. G. Bock, M. Diehl, D. B. Leineweber, and J. P. Schlöder, "A direct multiple shooting method for real-time optimization of nonlinear dae processes," in *Nonlinear Model Predictive Control*, F. Allgöwer and A. Zheng, Eds., Basel, Switzerland: Birkhäuser, 2000, pp. 245–267. Q12

[6] E. Bradford and L. Imsland, "Stochastic nonlinear model predictive control using Gaussian processes," in *Proc. Eur. Control Conf.*, 2018, pp. 1027–1034.

[7] K. Kassara and A. Moustafid, "Angiogenesis inhibition and tumor-immune interactions with chemotherapy by a control set-valued method," *Math. Biosci.*, vol. 231, no. 2, pp. 135–143, 2011.

[8] L. Magni and R. Scattolini, *Robustness and Robust Design of MPC for Nonlinear Discrete-Time Systems*. Berlin, Germany: Springer, 2007, pp. 239–254. [Online]. Available: https://doi.org/10.1007/978-3-540-72699-9_19

[9] D. Q. Mayne, J. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, pp. 789–814, 2000.

[10] D. Mayne, "Robust and stochastic model predictive control: Are we going in the right direction?," *Annu. Rev. Control*, vol. 41, pp. 184–192, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1367578816300098

[11] A. Mesbah, S. Streif, R. Findeisen, and R. Braatz, "Stochastic nonlinear model predictive control with probabilistic constraints," in *Proc. Amer. Control Conf.*, Jun. 2014, pp. 2413–2419.

[12] A. Mesbah, "Stochastic model predictive control: An overview and perspective for future research," *IEEE Control Syst. Mag.*, vol. 36, no. 6, pp. 30–44, Dec. 2016.

[13] A. Mesbah, "Stochastic model predictive control with active uncertainty learning: A survey on dual control," *Annu. Rev. Control*, vol. 45, pp. 107–117, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1367578817301232

[14] Z. K. Nagy and R. D. Braatz, "Distributional uncertainty analysis using power series and polynomial chaos expansions," *J. Process Control*, vol. 17, no. 3, pp. 229–240, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0959152406001119

[15] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

[16] T. Rigaut, P. Carpentier, J. Chancelier, M. D. Lara, and J. Waeytens, "Stochastic optimization of braking energy storage and ventilation in a subway station," *IEEE Trans. Power Syst.*, vol. 34, no. 2, pp. 1256–1263, Mar. 2019.

[17] G. Schildbach, L. Fagiano, C. Frei, and M. Morari, "The scenario approach for stochastic model predictive control with bounds on closed-loop constraint violations," *Automatica*, vol. 50, no. 12, pp. 3009–3018, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109814004166

[18] D. Xu and Y. Tian, "A comprehensive survey of clustering algorithms," *Ann. Data Sci.*, vol. 2, no. 2, pp. 165–193, Jun. 2015.