Data-driven distributionally robust iterative risk-constrained model predictive control

Zolanvari, Alireza; Cherukuri, Ashish

Published in:
2022 European Control Conference, ECC 2022

DOI:
10.23919/ECC55457.2022.9838319

Publication date:
2022

2022 European Control Conference (ECC)
July 12-15, 2022. London, United Kingdom

Data-driven distributionally robust iterative risk-constrained model predictive control

Alireza Zolanvari Ashish Cherukuri

Abstract— **This paper considers a risk-constrained infinite-horizon optimal control problem and proposes to solve it in an iterative manner. Each iteration of the algorithm generates a trajectory from the starting point to the target equilibrium state by implementing a distributionally robust risk-constrained model predictive control (MPC) scheme. At each iteration, a set of safe states (that satisfy the risk-constraint with high probability) and a certain number of samples of the uncertainty governing the risk constraint are available. These states and samples are accumulated in previous iterations. The safe states are used as terminal constraint in the MPC scheme and samples are used to construct a set of distributions, termed ambiguity set, such that it contains the underlying distribution of the uncertainty with high probability. The risk-constraint in each iteration is required to hold for all distributions in the ambiguity set. We establish that the trajectories generated by our iterative procedure are feasible, safe, and converge asymptotically to the equilibrium. Simulation example illustrates our results for the case of finding a risk-constrained path for a mobile robot in the presence of an uncertain obstacle.**

I. Introduction

Practical control systems often operate in uncertain environments, for example, a mobile robot navigating in the presence of obstacles. Safe optimal control in such situations can be modeled in many different ways. On the one hand, robust approaches consider the worst-case effect of the uncertainty on control design. On the other hand, popular probabilistic approaches model safety as chance-constraints in the optimal control problem and design deterministic or sample-based algorithms to solve it. A convenient strategy to balance these approaches is to consider appropriate risk constraints. We adopt this approach in our work and define a risk-constrained infinite-horizon optimal control problem. We assume that the task needs to be performed in an iterative way and the data regarding the uncertainty is incrementally revealed as iterations progress. We design an iterative method that combines the notions of learning model predictive control [1] and distributionally robust risk constraints [2].

Literature review: Optimization problems with worst-case expectation over a set of distributions, either in objective or constraints, is commonly termed as distributionally robust (DR) optimization [3]. The considered set of distributions is referred to as the ambiguity set. The DR framework is particularly attractive when the data regarding uncertainty is less. In this case, the decision-maker can construct the ambiguity set of appropriate size to tune the out-of-sample performance. Thus, DR optimization lends itself as a fitting tool for ensuring safety in uncertain systems. With this motivation, works [4]–[7] explore distributional robustness in model predictive control (MPC). Further, recent works [8], [9] investigate risk-averse MPC for Markovian switched systems, while making use of the connection that the so-called coherent risk measure of a random variable is equivalent to the worst-case expectation over a set of distributions.

While most of the above-listed works on MPC consider stochastic systems, we only focus on uncertain environments. This setup finds application in risk-averse motion planning, where our work is related to [10], [11]. Here risk constraints encode safety against collision. When only few samples regarding the uncertainty are available, [12], [13] use distributional robustness to ensure safety. However, none of these works explore the possibility of executing the task in an iterative manner. Such a method is appealing when data regarding the uncertainty is scarce at the beginning and more samples get revealed as the task is done repeatedly. As a consequence, the environment can be explored progressively. To actualize such a method, we make use of the learning model predictive framework introduced in [1]. Here, at each iteration, a part of the state space is explored and stored for future iterations where these states are used as terminal constraints. In [14], a learning-based MPC has been developed to tackle the uncertainties in the problem's constraints in a safe procedure. However, these strategies aim at satisfying robust and not risk constraints.

Setup and contributions: We define an infinite-horizon optimal control problem for a discrete-time deterministic system, where the state is subjected to a conditional value-at-risk constraint. The goal is to take the state from a starting point to a target equilibrium. Our main contribution is the design of the distributionally robust iterative MPC scheme that progressively approximates the solution of the infinite-horizon problem. In our procedure, at each iteration, we generate a trajectory using an MPC scheme, where a DR constrained finite-horizon problem is solved repeatedly. We assume a general class of ambiguity sets that are defined using the data collected in previous iterations. The terminal constraint in the finite-horizon problem enforces the state to lie in a subset of the safe states sampled in previous iterations. Once a trajectory is generated, the samples of the uncertainty collected in the iteration are added to the dataset and the sampled safe set is updated appropriately.

We establish three properties for our method. Under the assumption that a robustly feasible trajectory is available at the first iteration, we show that each iteration is recursively feasible and safe, where safety means satisfying the risk-

The authors are with the Engineering and Technology Institute Groningen, University of Groningen. Email: {a.zolanvari,a.k.cherukuri}@rug.nl. This work was partly supported with a scholarship from the Data Science and Systems Complexity (DSSC) Center, University of Groningen.

constraint with high probability. We prove that each trajectory asymptotically converges to the target state. Lastly, we give conditions under which the set of safe states grows and the cost of trajectory decreases as iterations progress. We apply our algorithm for finding a risk-averse path for a mobile robot in the presence of an uncertain obstacle [1].

II. Problem Statement

Consider the following discrete-time system:

$$x_{t+1} = f(x_t, u_t), \tag{1}$$

where $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ defines the dynamics and $x_t \in \mathbb{R}^{n_x}$ and $u_t \in \mathbb{R}^{n_u}$ are the state and control input of the system at time t, respectively. The system state and control input are subject to the following deterministic constraints: $x_t \in \mathcal{X}$, $u_t \in \mathcal{U}$ for all $t \geq 0$, where $\mathcal{X}$ and $\mathcal{U}$ are assumed to be *compact convex* sets. The aim is to solve an infinite-horizon risk-constrained optimal control problem for system (1) that drives the system to a target equilibrium point $x_F \in \mathcal{X}$. To that end, let $r : \mathcal{X} \times \mathcal{U} \to \mathbb{R}_{\geq 0}$ be a continuous function that represents the *stage cost* associated to the optimal control problem. We assume that $r(x, u) \geq 0$ for all $(x, u) \in \mathcal{X} \times \mathcal{U}$ and $r(x, u) = 0$ if and only if $(x, u) = (x_F, 0)$. The risk-constrained infinite-horizon optimal control problem is given as

$$\begin{aligned}
\min \quad & \textstyle\sum_{t=0}^{\infty} r(x_t, u_t) && \text{(2a)}\\
\text{s.t.} \quad & x_{t+1} = f(x_t, u_t), \quad \forall t \geq 0, && \text{(2b)}\\
& x_t \in \mathcal{X}, u_t \in \mathcal{U}, \quad \forall t \geq 0, && \text{(2c)}\\
& x_0 = x_S, && \text{(2d)}\\
& \mathrm{CVaR}_{\beta}^{\mathbb{P}}\left[g(x_t, w)\right] \leq \delta, \quad \forall t \geq 0, && \text{(2e)}
\end{aligned}$$

where $x_S \in \mathcal{X}$ is the initial state and constraint (2e) represents the risk-averseness. Here, CVaR stands for the conditional value-at-risk [2], w is a random variable with distribution $\mathbb{P}$ supported on the compact set $\mathcal{W} \subset \mathbb{R}^{n_w}$, $\delta > 0$ is the risk tolerance parameter, $\beta > 0$ is the risk-averseness coefficient, and the continuous function $g : \mathcal{X} \times \mathcal{W} \to \mathbb{R}$ is referred to as the constraint function. The constraint (2e) ensures that the risk associated to the state at any time, as specified using the random function g, is bounded by a given parameter δ. More generally, the constraint can be perceived as a safety specification for system (1) under uncertain environments.

The infinite-horizon problem (2) is difficult to solve in general due to state, input, and risk constraints. Besides, in practice, the distribution $\mathbb{P}$ is usually unknown beforehand. To tackle these challenges, we propose a data-driven iterative MPC scheme outlined in the following section.

[1]We use the following notation throughout. Let $\mathbb{R}$, $\mathbb{R}_{\geq 0}$, and $\mathbb{N}$ denote the set of real, non-negative real, and natural numbers, resp. The set of natural numbers excluding zero is denoted as $\mathbb{N}_{\geq 1}$. Let $\|\cdot\|$ and $\|\cdot\|_1$ denote the Euclidean 2- and 1-norm, resp. For $N \in \mathbb{N}$, we denote $[N] := \{0, 1, \ldots, N\}$. Given $x \in \mathbb{R}$, we let $[x]_+ = \max(x, 0)$. Given two sets X and Y, a set-valued map $f : X \rightrightarrows Y$ associates to each point in X a subset of Y. The n-fold Cartesian product of a set $\mathcal{S}$ is denoted as $\mathcal{S}^n$. The n-dimensional unit simplex is denoted as Δ_n.

[2]Given a real-valued random variable Z with probability distribution $\mathbb{P}$ and parameter $\beta \in (0, 1)$, the *conditional value-at-risk (CVaR)* of Z at level β, denoted $\mathrm{CVaR}_{\beta}^{\mathbb{P}}[Z]$, is given as [15], $\mathrm{CVaR}_{\beta}^{\mathbb{P}}[Z] = \inf_{t \in \mathbb{R}} \Big\{ t + \beta^{-1}\mathbb{E}^{\mathbb{P}}[Z - t]_+ \Big\}$, where $\mathbb{E}^{\mathbb{P}}[\cdot]$ denotes expectation under $\mathbb{P}$. The parameter β characterizes risk-averseness. When β is close to unity, the decision-maker is risk-neutral, whereas, β close to the origin implies high risk-averseness.

III. Distributionally Robust Risk-constrained Iterative MPC

In this section, we provide an iterative strategy for solving the infinite-horizon optimal control problem (2) in an approximate manner. Here, each iteration refers to an execution of the control task, that is, taking the system state from x_S to x_F in a safe manner. Our iterative framework is inspired by [1] and roughly proceeds in the following manner. At the start of any iteration j, we have access to a finite number of samples of the uncertainty, a set of safe states, and the cost it takes to go from each of these safe states to the target. In iteration j, we use this prior knowledge and define an MPC scheme that constructs a safe trajectory starting at x_S and ending at x_F. The aim of this newly generated trajectory is to possibly reduce the cost or improve the safety as compared to the previous iterations. At the end of the iteration, we update the dataset with samples gathered along the execution of the MPC scheme. Subsequently, we update the set of safe states. In the following, we make all the necessary ingredients of the iterative framework precise and later put them together in the form of Algorithm 1.

A. Components of the Iterative Framework

1) Trajectories: Every iteration results into a trajectory. The system state and the control input at time t of the j^{th} iteration are denoted as x_t^j and u_t^j, respectively, and the j^{th} *trajectory* is given by concatenated sets:

$$\begin{aligned}
\mathsf{x}^j &:= [x_0^j, x_1^j, \ldots, x_t^j, \ldots, x_{T_j}^j],\\
\mathsf{u}^j &:= [u_0^j, u_1^j, \ldots, u_t^j, \ldots, u_{T_j-1}^j].
\end{aligned} \tag{3}$$

We assume that all trajectories start from x_S, that is, $x_0^j = x_S$ for all $j \geq 1$. While our objective is to solve an infinite-horizon problem (2), for practical considerations, we aim to find trajectories that reach the target x_F in a finite number of steps. Thus, we assume that for each iteration j, the *length* of the trajectory is finite, denoted by $T_j \in \mathbb{N}_{\geq 1}$. Throughout the paper, whenever we mention trajectory of states, we implicitly mean that there exists a feasible control sequence that makes this trajectory of states possible.

2) Data and Ambiguity Sets: At the start of iteration j, a *dataset* $\widehat{\mathcal{W}}^{j-1} := \{\widehat{w}_1, \ldots, \widehat{w}_{N_{j-1}}\} \subset \mathcal{W}$ of N_{j-1} i.i.d. samples of the uncertainty w drawn from $\mathbb{P}$ is available. Here, the index $j - 1$ indicates the samples collected till iteration $j - 1$. We assume that we collect one sample per time-step of each iteration and so the number of samples available for iteration $j+1$ are $N_j = N_{j-1} + T_j$. Our aim is to use the dataset $\widehat{\mathcal{W}}^{j-1}$ to enforce the risk constraint (2e) in an appropriate sense for the trajectory generated in the j^{th} iteration. To this end, we adopt a distributionally robust approach. That is, we generate a set of distributions, termed *ambiguity set*, that contains the underlying distribution $\mathbb{P}$ with high probability. We then enforce the risk constraint (2e) for all distributions in the ambiguity set. To put the notation in

place, assume that given a *confidence parameter* $\zeta \in (0,1)$, we have access to a map $\mathbb{D} : \mathcal{W}_\infty \rightrightarrows \mathcal{P}(\mathcal{W})$ such that given any set of N i.i.d samples $\widehat{\mathcal{W}}_N = \{\widehat{w}_1, \ldots, \widehat{w}_N\}$ the set of distributions $\mathbb{D}(\widehat{\mathcal{W}}_N)$ contains $\mathbb{P}$ with confidence ζ. In the definition of the map, the domain is $\mathcal{W}_\infty = \cup_{i=1}^{\infty} \mathcal{W}^i$ and $\mathcal{P}(\mathcal{W})$ denotes the set of all distributions supported on $\mathcal{W}$. We assume that $\mathbb{D}$ always leads to a closed and nonempty ambiguity set. We term $\mathbb{D}$ as the *ambiguity set generating map*. Given $\mathbb{D}$, our strategy is to set the ambiguity set used for iteration j as $\mathcal{D}^{j-1} := \mathbb{D}(\widehat{\mathcal{W}}^{j-1})$. The assumption on $\mathbb{D}$ imply that $\mathcal{D}^{j-1}$ is $(\zeta, \mathbb{P}^{|\widehat{\mathcal{W}}^{j-1}|})$-*reliable*, that is,

$$\mathbb{P}^{|\widehat{\mathcal{W}}^{j-1}|}\left(\mathbb{P} \in \mathcal{D}^{j-1}\right) \geq \zeta. \tag{4}$$

The above property implies that for the MPC scheme related to the j^{th} iteration, if we impose the risk constraint (2e) for all distributions in $\mathcal{D}^{j-1}$, then the generated trajectory will satisfy the risk constraint with at least probability ζ. Ideally, we must aim to find trajectories that satisfy (2e). However, when only limited data regarding the uncertainty is known, one can only enforce such a constraint in a probabilistic manner and the above definition aims to capture this feature.

3) Cost-to-go: The cost-to-go from time t for the trajectory $(\mathsf{x}^j, \mathsf{u}^j)$ generated in iteration j, is denoted as:

$$J^j_{(t:\infty)} := \textstyle\sum_{k=t}^{\infty} r(x^j_k, u^j_k). \tag{5}$$

Setting $t = 0$ in (5) gives us the cost of the j^{th} iteration as $J^j_{(0:\infty)}$, that measures the performance of the controller in that iteration. For every time-step $t \geq T_j$, we assume that the system remains at x_F and the control input is zero. Thus, the infinite sum in (5) is well-defined as $r(x_F, 0) = 0$.

4) Sampled safe set: The main advantage of the iterative scheme is that it allows data to be gathered and state-space to be explored in an incremental manner. That is, we keep track of all samples from past iterations (discussed above) and we also maintain a set of safe states (along with their respective minimum cost-to-go) that were visited in the previous iterations. These safe states are form terminal constraints in the MPC scheme (as proposed in [1]). In iteration j, the risk constraint (2e) is imposed for all distributions in $\mathcal{D}^{j-1}$ in the finite-horizon optimal control problem solved in the MPC scheme (see Section III-A.6). Thus, due to (4), the trajectory $(\mathsf{x}^j, \mathsf{u}^j)$ is $(\zeta, \mathbb{P}^{|\widehat{\mathcal{W}}^{j-1}|})$-*safe*, that is

$$\mathbb{P}^{|\widehat{\mathcal{W}}^{j-1}|}\left(\mathrm{CVaR}^{\mathbb{P}}_{\beta}\left[g(x^j_t, w)\right] \leq \delta\right) \geq \zeta \tag{6}$$

for all $t \in [T_j]$. Note that x^j is safe with respect to the dataset $\widehat{\mathcal{W}}^{j-1}$. However, since the next iteration $j+1$ is built considering safety with respect to the dataset $\widehat{\mathcal{W}}^{j}$, all previously generated trajectories need to be $(\zeta, \mathbb{P}^{|\widehat{\mathcal{W}}^{j}|})$-safe to be considered as the set of safe states in iteration $j+1$. In particular, the *sampled safe set* obtained at the end of iteration j and to be used in iteration $j+1$, denoted $\mathcal{S}^j \subseteq [\,j\,] \times \mathcal{X} \times \mathbb{R}_{\geq 0}$, is defined recursively as

$$\mathcal{S}^j = \mathbb{S}^j\Big(\mathcal{S}^{j-1} \cup \{(j, x^j_t, J^j_{(t:\infty)})\}_{t=1}^{T_j}\Big). \tag{7}$$

In the above expression, the set $\{(j, x^j_t, J^j_{(t:\infty)})\}_{t=1}^{T_j}$ collects the set of states visited in iteration j, along with the associated cost-to-to. The counter j is maintained in this set to identify the iteration to which a state with a particular cost-to-go is associated with. The set $\mathcal{S}^{j-1}$ is the sampled safe set used in iteration j. The map $\mathbb{S}^j$ only keeps the states that are safe with respect to the new data set $\widehat{\mathcal{W}}^j$. This aspect of our method is different from [1] where explored sates are safe for all future iterations. The exact procedure that defines $\mathbb{S}^j$ is given in our algorithm.

For ease of exposition, we define maps $\Pi_{\text{traj}}(\cdot)$, $\Pi_{\text{state}}(\cdot)$, and $\Pi_{\text{cost}}(\cdot)$, such that, given a safe set $\mathcal{S}$, $\Pi_{\text{traj}}(\mathcal{S})$, $\Pi_{\text{state}}(\mathcal{S})$, and $\Pi_{\text{cost}}(\mathcal{S})$ return the set of all trajectory indices, states, and cost-to-go values that appear in $\mathcal{S}$, respectively. The following assumption is required to initialize our iterative procedure with a nonempty sampled safe set.

Assumption III.1. *(Initialization):* Before starting the first iteration, the sampled safe set $\mathcal{S}^0$ contains the states of a finite-length robustly safe trajectory x^0 that starts from x_S and reaches x_F. That is, $x \in \mathcal{X}$, $g(x, w) \leq \delta$ for all $w \in \mathcal{W}$, and all $x \in \Pi_{\text{state}}(\mathcal{S}^0)$. •

5) Minimum Cost-to-go: The sampled safe set $\mathcal{S}^j$ keeps track of the cost-to-go associated with each state in the set. However, a state can appear in more than one trajectory. For such cases, we need to maintain the minimum cost-to-go associated with a state. To this end, given $\mathcal{S}^j$, we define the associated minimum cost-to-go map as

$$Q^j(x) := \begin{cases} \min\limits_{J \in F^j(x)} J, & x \in \Pi_{\text{state}}(\mathcal{S}^j), \\ +\infty, & x \notin \Pi_{\text{state}}(\mathcal{S}^j), \end{cases} \tag{8}$$

where $F^j(x) = \{J^i_{(t:\infty)} \mid \Pi_{\text{state}}(\{(i, x^i_t, J^i_{(t:\infty)})\}) = \{x\}, (i, x^i_t, J^i_{(t:\infty)}) \in \mathcal{S}^j\}$. Here, the set $F^j(x)$ collects all cost-to-go values associated to the state $x \in \Pi_{\text{state}}(\mathcal{S}^j)$. The function Q^j then finds the minimum among these.

6) DR Risk-constrained Finite-Horizon Problem: We now present the finite-horizon optimal control problem solved at each time-step of each iteration. We write the problem for generic current state x, sampled safe set $\overline{\mathcal{S}}$, and ambiguity set $\overline{\mathcal{D}}$. Let $K \in \mathbb{N}_{\geq 1}$ be the length of the horizon and consider

$$\mathcal{J}_{(\overline{\mathcal{S}}, \overline{\mathcal{D}})}(x) := \begin{cases} \min & \sum_{k=0}^{K-1} r(x_k, u_k) + \overline{Q}(x_K) \\ \text{s.t.} & x_{k+1} = f(x_k, u_k), \forall k \in [K-1], \\ & x_k \in \mathcal{X}, u_k \in \mathcal{U}, \forall k \in [K-1], \\ & x_0 = x,\ x_K \in \Pi_{\text{state}}(\overline{\mathcal{S}}), \\ & \sup_{\mu \in \overline{\mathcal{D}}} \left[\mathrm{CVaR}^{\mu}_{\beta}\left[g(x_k, w)\right]\right] \leq \delta, \\ & \qquad\qquad \forall k \in [K-1], \end{cases} \tag{9}$$

where $\overline{Q} : \mathcal{X} \to \mathbb{R}$ gives the minimum cost-to-go for all states in $\overline{\mathcal{S}}$ and is calculated in a similar manner as in (8). The decision variables in the above problem are $(x_0, x_1, \ldots, x_K)$ and $(u_0, u_1, \ldots, u_{K-1})$. The set $\overline{\mathcal{S}}$ defines the terminal constraint $x_K \in \Pi_{\text{state}}(\overline{\mathcal{S}})$. Finally, note that the risk constraint is required to hold for all distributions in

the ambiguity set $\overline{\mathcal{D}}$. Thus, we refer to it as *distributionally robust (DR) constraint*. For iteration j and time-step t, the MPC scheme solves the finite-horizon problem (9) with $x = x_t^j$, $\overline{\mathcal{S}} = \mathcal{S}^{j-1}$, $\overline{\mathcal{D}} = \mathcal{D}^{j-1}$, and $\overline{Q} = Q^{j-1}$.

B. The Iterative Framework

Here, we compile the elements described in the previous section and present our iterative procedure termed *distributionally robust risk-constrained iterative MPC (DR-RC-Iterative-MPC)*. The informal description is given below.

[Informal description of Algorithm 1]: Each iteration $j \geq 1$ starts with a sampled safe set $\mathcal{S}^{j-1}$ and an ambiguity set $\mathcal{D}^{j-1}$. The ambiguity set is constructed (see Line 3) using samples in dataset $\widehat{\mathcal{W}}^{j-1}$ collected in previous iterations and the map $\mathbb{D}$ that ensures (4). In the first step of the iteration (Line 2), a trajectory $(\mathsf{x}^j, \mathsf{u}^j)$ is generated by the DR_MPC routine (described in Algorithm 2) to which the sampled safe set $\mathcal{S}^{j-1}$ and the ambiguity set $\mathcal{D}^{j-1}$ are given as inputs. This trajectory is $(\zeta, \mathbb{P}^{|\widehat{\mathcal{W}}^{j-1}|})$-safe, that is, it satisfies (6). The samples collected in iteration j are appended to the dataset $\widehat{\mathcal{W}}^{j-1}$ in Line 2 and the ambiguity set for the next iteration is constructed in Line 3. The trajectory x^j along with its associated cost-to-go is appended to the sampled safe set in Line 4. In Lines 5 to 7, the sampled safe set $\mathcal{S}^{j-1}$ is modified to make sure that it only contains trajectories that are $(\zeta, \mathbb{P}^{|\widehat{\mathcal{W}}^{j}|})$-safe. These steps collectively represent the map $\mathbb{S}$ defined in an abstract manner in (7). The indices of trajectories present in $\mathcal{S}^{j-1}$ are maintained in the set $\mathcal{I}^{j-1}$. In Line 5, trajectories in $\mathcal{I}^{j-1} \cup \{j\}$ for which at least one state is not $(\zeta, \mathbb{P}^{|\widehat{\mathcal{W}}^{j}|})$-safe are enumerated in the set $\mathcal{UI}^j$. Accordingly, in Line 6, the set $\mathcal{I}^j$ is updated as trajectories in $\mathcal{I}^{j-1} \cup \{j\}$ that are not in $\mathcal{UI}^j$. The states visited in these trajectories are stored in $\mathcal{S}^j$ in Line 7. Finally, the minimum cost-to-go for all states in $\mathcal{S}^j$ is updated in Line 8

Note that in the above algorithm, the sampled safe set is updated in an iterative way. That is, we add the j^{th} trajectory to $\mathcal{S}^{j-1}$ and then check safety with respect to the dataset $\widehat{\mathcal{W}}^j$. In the process, we loose some trajectories in iterations $\{1, \ldots, j-1\}$ that could have been $(\zeta, \mathbb{P}^{|\widehat{\mathcal{W}}^{j}|})$-safe.

Algorithm 1 calls the DR_MPC routine in each iteration to generate the trajectory. This procedure is given in Algorithm 2 and informally described below.

[Informal description of Algorithm 2]: The procedure generates a trajectory from x_S to x_F given a sampled safe set $\overline{\mathcal{S}}$ and an ambiguity set $\overline{\mathcal{D}}$. The function $\overline{Q}$ is computed using (8). At time-step t, problem (9) is solved with $x = x_t$ giving solution

$$\begin{aligned} x^*_{\text{vec},t} &= [x^*_{t|t}, \ldots, x^*_{t+K|t}], \\ u^*_{\text{vec},t} &= [u^*_{t|t}, \ldots, u^*_{t+K-1|t}], \end{aligned} \tag{10}$$

where $x_{t+k|t}$ is the prediction made at t regarding the state at $t+k$. The control action at time t is set as the first element of $u^*_{\text{vec},t}$ (Line 5) and it is appended to the trajectory u. The state is updated and added to x in Line 6. The procedure moves to the next time step with the updated state as x_{t+1}.

Algorithm 1: DR-RC-Iterative-MPC

Input : $\mathcal{S}^0$ – Initial sampled safe set
$\widehat{\mathcal{W}}^0$ – Initial set of samples
$\mathcal{I}^0$ – Index of trajectory in $\mathcal{S}^0$
Initialize : $j \leftarrow 1$, $\mathcal{D}^0 = \mathbb{D}(\widehat{\mathcal{W}}^0)$, $\mathcal{UI}^0 \leftarrow \emptyset$

1 **while** $j > 0$ **do**
2 Set $(\mathsf{x}^j, \mathsf{u}^j) \leftarrow$ DR_MPC$(\mathcal{S}^{j-1}, \mathcal{D}^{j-1})$; $T^j \leftarrow$ length(x^j); $\widehat{\mathcal{W}}^j \leftarrow \widehat{\mathcal{W}}^{j-1} \cup \{\widehat{w}_i\}_{i=1}^{T^j}$
3 Set $\mathcal{D}^j \leftarrow \mathbb{D}(\widehat{\mathcal{W}}^j)$
4 Set $\mathcal{S}^{j-1} \leftarrow \mathcal{S}^{j-1} \cup \{(j, x_t^j, J_{t\to\infty}^j)\}_{t=1}^{T_j}$
5 Set $\mathcal{UI}^j \leftarrow \{i \in (\mathcal{I}^{j-1} \cup \{j\}) \,|\, (i, x, J) \in \mathcal{S}^{j-1}, \sup_{\mu \in \mathcal{D}^j} \left[\text{CVaR}_\beta^\mu [g(x, w)]\right] > \delta\}$
6 Set $\mathcal{I}^j \leftarrow (\mathcal{I}^{j-1} \cup \{j\}) \setminus \mathcal{UI}^j$
7 Set $\mathcal{S}^j \leftarrow \{(i, x, J) \in \mathcal{S}^{j-1} \mid i \in \mathcal{I}^j\}$
8 Compute $Q^j(x)$ for all $x \in \Pi_{\text{state}}(\mathcal{S}^j)$ using (8)
9 Set $j \leftarrow j + 1$

Algorithm 2: Distributionally robust MPC function

Function DR_MPC $(\overline{\mathcal{S}}, \overline{\mathcal{D}})$:
Initialize : $t \leftarrow 0$; $x_0 \leftarrow x_S$; $\mathsf{x} \leftarrow [x_0]$, $\mathsf{u} \leftarrow [\,]$
Set $\overline{Q}$ as minimum cost-to-go in $\overline{\mathcal{S}}$ (use (8))
while $x_t \neq x_F$ **do**
Solve (9) with $x = x_t$ and obtain optimal solutions $x^*_{\text{vec},t}$ and $u^*_{\text{vec},t}$
Set $u_t \leftarrow u^*_{t|t}$; $\mathsf{u} \leftarrow [u, u_t]$
Set $x_{t+1} \leftarrow f(x_t, u_t)$; $\mathsf{x} \leftarrow [x, x_{t+1}]$; $t \leftarrow t+1$
return (x, u)

The above explained MPC procedure might not terminate in finite time, thus possibly violating our assumption that all trajectories have finite length. To practically overcome this issue, we terminate the MPC scheme when the state reaches a neighborhood of the equilibrium x_F.

Remark III.1. *(Tractability):* Note that, if $g(\cdot, w)$ is convex for every $w \in \mathcal{W}$ and (1) is a linear system, then the constraint (2e) as well as the DR risk-constraint in (9) are convex. Due to the latter fact, all points in the convex hull of $\Pi_{\text{state}}(\overline{\mathcal{S}})$ satisfy the DR risk-constraint. Hence, we can replace $\Pi_{\text{state}}(\overline{\mathcal{S}})$ with its convex hull and define the minimum cost-to-go function using Barycentric functions (see [16]) in the problem (9) without affecting the safety of the resulting trajectory. By doing so, problem (9) becomes convex which makes it computationally efficient to solve. •

Remark III.2. *(Ambiguity sets):* The ambiguity set in our algorithm is defined by an arbitrary map $\mathbb{D}$. Popular choices of data-based ambiguity sets are the ones using distance metrics such as Wasserstein, KL-divergence, ϕ-divergence or using moment information, see [3] for a survey. •

Remark III.3. *(Safety vs cost-performance):* The reliability parameter ζ in our framework is tunable. Meaning, if one requires high level of safety, then ζ can be selected close to unity. In that case, the ambiguity set needs to be large enough to ensure (4) and so the DR risk-constraint turns out to be conservative. Analogously, if cost improvement is the goal, then a low value of ζ will be sufficient. •

IV. Properties of DR-RC-Iterative-MPC

We first present the recursive feasibility and the safety guarantee of the trajectories generated by our iterative procedure given in Algorithm 1.

Proposition IV.1. *(Safety and recursive feasibility of DR-RC-Iterative-MPC): Let Assumption III.1 hold. Then, at each iteration $j \geq 1$ and time-step $t \geq 0$, the finite-horizon problem* (9) *with $x = x_t^j$, $\overline{\mathcal{S}} = \mathcal{S}^{j-1}$, and $\overline{\mathcal{D}} = \mathcal{D}^{j-1}$ solved in the DR-RC-Iterative-MPC scheme is feasible. Further, the generated trajectory $(\mathsf{x}^j, \mathsf{u}^j)$ is $(\zeta, \mathbb{P}^{\widehat{\mathcal{W}}^{j-1}})$-safe.* •

Next, we give the asymptotic convergence of each trajectory generated by Algorithm 2.

Proposition IV.2. *(Convergence of* DR_MPC*): Let Assumption III.1 hold. Then, for each iteration $j \geq 1$ of the DR-RC-Iterative-MPC procedure, the trajectory $(\mathsf{x}^j, \mathsf{u}^j)$ generated by* DR_MPC *satisfies $x_t^j \to x_F$ as $t \to \infty$.* •

Next, we examine the performance across iterations.

Proposition IV.3. *(Guarantee across iterations for DR-RC-Iterative-MPC): For the DR-RC-Iterative-MPC procedure, if $\mathcal{D}^j \subset \mathcal{D}^{j-1}$ for some iteration $j \geq 1$, then we have $\mathcal{S}^j = \mathcal{S}^{j-1} \cup \{(j, x_t^j, J_{(t:\infty)}^j)\}_{t=1}^{T_j}$. As a consequence, $\mathcal{S}^{j-1} \subseteq \mathcal{S}^j$. In addition, if the function $\mathcal{J}_{(\mathcal{S}^{j-1}, \mathcal{D}^{j-1})}$ is continuous at x_F, then $J_{(0:\infty)}^j \leq J_{(0:\infty)}^{j-1}$.* •

V. Simulation

We demonstrate the performance of Algorithm 1 via a motion planning task for a mobile robot where the environment includes a randomly moving obstacle.

1) Setup: Consider the following model for a circular mobile robot navigating in a 2D environment:

$$x_{t+1} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} u_t.$$

Here, the state $x = [z, y, v_z, v_y]^\top$ contains the position (z, y) of the center of mass of the robot and its velocity in z and y directions. The input $u = [a_z, a_y]^\top$ consists of the acceleration in z and y directions. The objective of this problem is to steer the agent from the initial point $x_S = [0, 0, 0, 0]^\top$ to the target point $x_F = [5, 3, 0, 0]^\top$ while constraining the risk of colliding with a square obstacle of length $\ell_{\mathcal{O}} = 0.4$ that moves randomly around the point $[2, 2]^\top$. Specifically, the position of the obstacle in each time-step is given by $o_t = \begin{bmatrix} 2 & 2 \end{bmatrix}^\top + \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}^\top w_t$, where $w_t \in \mathbb{R}$ is the uncertainty defined by the Beta-binomial$(15, 10, 15)$ distribution supported on the set of fifteen points $\{-0.5 + i/14 \mid i \in [14]\}$. Since a small number of samples are usually available in practice, we start with only $N_0 = 5$ samples. We assume that in each time-step of each iteration, the obstacle's position is observable, which forms the dataset of samples. Given position o_t, the region of the environment occupied by the obstacle is given as

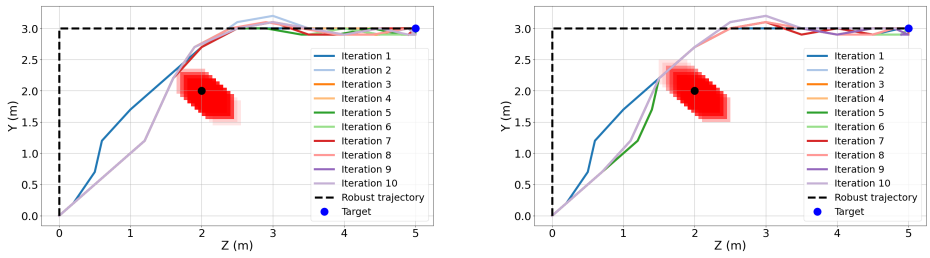
$$\mathcal{O}_t = \{(z, y) \in \mathbb{R}^2 \mid o_t - \tfrac{\ell_{\mathcal{O}}}{2} \mathbf{1}_2 \leq [z \; y]^\top \leq o_t + \tfrac{\ell_{\mathcal{O}}}{2} \mathbf{1}_2\},$$

where $\mathbf{1}_2 = [1, 1]^\top$. The stage cost is quadratic, given as $r(x_t, u_t) = (x_F - x_t)^\top Q (x_F - x_t) + u_t^\top R u_t$, where $Q = \text{diag}(1, 1, 0.01, 0.01)$ and $R = \text{diag}(0.01, 0.01)$. The safe set $\mathcal{S}^0$ is generated using an open-loop control, see dashed-black line in Figure 1. We execute the algorithm for 20 iterations. The prediction horizon is $K = 5$. We use $\beta = 0.05$ as the risk-averseness coefficient and $\delta = 0.02$ as the right-hand side of the risk constraint.

We consider ambiguity sets defined using the total variation distance. For discrete distributions $P, Q \in \Delta_{|\mathcal{W}|}$ supported on a finite set $\mathcal{W}$, the total variation distance between them is defined as $\delta(P, Q) = \frac{1}{2}\|P - Q\|_1$. Given N i.i.d samples $\{\widehat{w}_1, \ldots, \widehat{w}_N\}$ of the uncertainty, the empirical distribution is given by the vector $\widehat{\mathbb{P}}^N := (p_i^N)_{i=1}^{|\mathcal{W}|}$, where $p_i^N = (\text{frequency of } w_i \in \mathcal{W} \text{ in the dataset})/N$. Using this definition, we consider the ambiguity sets of the form $\mathcal{D} = \{\mu \in \Delta_{|\mathcal{W}|} \mid \delta(\mu, \widehat{\mathbb{P}}^N) \leq \theta\}$, where $\theta \geq 0$ is the radius.

For collision-avoidance, the constraint function g is given as the distance between the agent and the safe region $\mathcal{Y}_t$ determined by excluding the instantaneous position of the obstacle from the environment. More precisely, $g(x, w) = \min_{a \in \mathcal{Y}} \|Cx - a\|$, where $\mathcal{Y} := \mathbb{R}^2 \backslash \mathcal{O}$, the set $\mathcal{O}$ is determined by the uncertainty w, and C is chosen such that $Cx = [z, y]^\top$. Taking benefit of the square shape of the obstacle, we use the simplified representation of function g provided in [12] which is $g(x, w) = \min_{j \in [3]} \left\{ \frac{\left[d_j + h_j^\top (Cx - w)\right]_+}{\|h_j\|} \right\}$, where h_j and d_j represent the outward normal and the position of one of the constraints defining the obstacle set (see [12, Lemma 1] for details). Note that g, and so the distributionally robust constraint (9) are non-convex, and non-trivial to handle. We reformulate this constraint into a finite-dimensional form, see the extended version [17] for details. The problem (9), considering the reformulation is implemented in GEKKO [18] using APOPT solver.

2) Results: The trajectories for different ambiguity set sizes are presented in Figure 1. The first iteration is same for all experiments, given by the robustly feasible trajectory. For small ambiguity sets, the trajectories are closer to the obstacle. For larger ones, the algorithm becomes more conservative to the extent that for $\theta = 0.5$ the agent stops exploring and is only concerned about safety. There is a noteworthy observation in Figure 1b, that is, trajectories get closer to the obstacle in the first few iterations but as more data is collected, the safe set gets refined in later iterations and the robot deviates from the obstacle more strongly. Finally in Figure 2 we underline the impact of the size of the ambiguity set cost-performance and safety. As shown, smaller ambiguity sets provide cost-efficient trajectories while increasing the probability of collision.



(a) $\theta = 5 \times 10^{-6}$ (b) $\theta = 5 \times 10^{-4}$

(c) $\theta = 5 \times 10^{-2}$ (d) $\theta = 0.5$

Fig. 1: Plots illustrating the application of the DR-RC-Iterative-MPC procedure for the task of navigating the mobile robot in an environment with an uncertain obstacle (see Section V for details). We consider four different radii for the ambiguity set and for each case, the radius does not change over the iterations. The initial robust trajectory (dashed black line) is the same for all cases. Each realization of the obstacle is plotted with a shaded red square. As observed, the trajectories become more conservative as the radius of the ambiguity set increases.

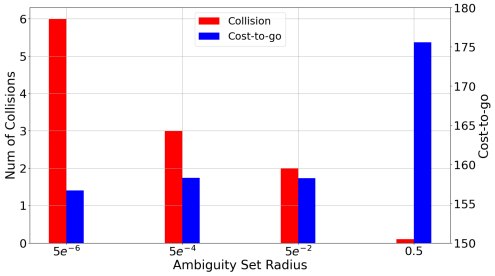


Fig. 2: The effect of the size of the ambiguity set on safety and performance. The red block represents the number of iterations (out of 20) in which the trajectory collides with the obstacle at least once. The blue block depicts the iteration cost of the collision-free iteration that has the highest index.

VI. Conclusions

We considered a risk-constrained infinite-horizon optimal control problem and designed an iterative MPC-based scheme to solve it. Our procedure approximated the risk constraints using their distributionally robust counterparts. Each iteration generated a trajectory that is safe and that converges to the equilibrium asymptotically. In future we wish to explore the finite-time convergence of the MPC scheme, the convergence of the iterative procedure, and a computationally tractable implementation for multiple agents.

References

[1] U. Rosolia and F. Borrelli, "Learning model predictive control for iterative tasks. a data-driven control framework," *IEEE Transactions on Automatic Control*, vol. 63, no. 7, pp. 1883–1896, 2017.

[2] A. R. Hota, A. Cherukuri, and J. Lygeros, "Data-driven chance constrained optimization under Wasserstein ambiguity sets," in *American Control Conference*, pp. 1501–1506, July 2019.

[3] H. Rahimian and S. Mehrotra, "Distributionally robust optimization: A review," 2019. arXiv preprint available at `https://arxiv.org/abs/1908.05659`.

[4] C. Mark and S. Liu, "Data-driven distributionally robust MPC: An indirect feedback approach," 2021. arXiv preprint available at `https://arxiv.org/abs/2109.09558`.

[5] J. Coulson, J. Lygeros, and F. Dorfler, "Distributionally robust chance constrained data-enabled predictive control," *IEEE Transactions on Automatic Control*, pp. 1–1, 2021.

[6] P. Coppens and P. Patrinos, "Data-driven distributionally robust MPC for constrained stochastic systems," *IEEE Control Systems Letters*, vol. 6, pp. 1274–1279, 2022.

[7] M. Schuurmans and P. Patrinos, "A general framework for learning-based distributionally robust MPC of Markov jump systems," 2021. arXiv preprint available at `https://arxiv.org/abs/2106.00561`.

[8] P. Sopasakis, D. Herceg, A. Bemporad, and P. Patrinos, "Risk-averse model predictive control," *Automatica*, vol. 100, pp. 281–288, 2018.

[9] S. Singh, Y. Chow, A. Majumdar, and M. Pavone, "A framework for time-consistent, risk-sensitive model predictive control: Theory and algorithms," *IEEE Transactions on Automatic Control*, vol. 64, no. 7, pp. 2905–2912, 2019.

[10] A. Hakobyan, G. C. Kim, and I. Yang, "Risk-aware motion planning and control using CVaR-constrained optimization," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3924–3931, 2019.

[11] A. Dixit, M. Ahmadi, and J. W. Burdick, "Risk-sensitive motion planning using entropic value-at-risk," 2020. arXiv preprint available at `https://arxiv.org/abs/2011.11211`.

[12] A. Hakobyan and I. Yang, "Wasserstein distributionally robust motion planning and control with safety constraints using conditional value-at-risk," in *IEEE Int. Conf. on Robotics and Automation*, pp. 490–496, IEEE, 2020.

[13] A. Hakobyan and I. Yang, "Distributionally robust risk map for learning-based motion planning and control: A semidefinite programming approach," 2021. arXiv preprint available at `https://arxiv.org/abs/2105.00657`.

[14] M. Bujarbaruah, C. Vallon, and F. Borrelli, "Learning to satisfy unknown constraints in iterative MPC," 2020. arXiv preprint available at `https://arxiv.org/abs/2006.05054`.

[15] A. Shapiro, D. Dentcheva, and A. Ruszczyński, *Lectures on stochastic programming*. Philadelphia, PA: SIAM, 2014.

[16] U. Rosolia and F. Borrelli, "Learning model predictive control for iterative tasks: A computationally efficient approach for linear system," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 3142–3147, 2017. 20th IFAC World Congress.

[17] A. Zolanvari and A. Cherukuri, "Data-driven distributionally robust iterative risk-constrained model predictive control," 2022. arXiv preprint available at `https://arxiv.org/abs/2111.12977`.

[18] L. Beal, D. Hill, R. Martin, and J. Hedengren, "Gekko optimization suite," *Processes*, vol. 6, no. 8, p. 106, 2018.