IEEE Control Systems Letters paper presented at
2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

An Improved Data Augmentation Scheme for Model Predictive Control Policy Approximation

Dinesh Krishnamoorthy *Senior Member*

Abstract— **This paper considers the problem of data generation for MPC policy approximation. Learning an approximate MPC policy from expert demonstrations requires a large data set consisting of optimal state-action pairs, sampled across the feasible state space. Yet, the key challenge of efficiently generating the training samples has not been studied widely. Recently, a sensitivity-based data augmentation framework for MPC policy approximation was proposed, where the parametric sensitivities are exploited to cheaply generate several additional samples from a single offline MPC computation. The error due to augmenting the training data set with inexact samples was shown to increase with the size of the neighborhood around each sample used for data augmentation. Building upon this work, this letter paper presents an improved data augmentation scheme based on predictor-corrector steps that enforces a user-defined level of accuracy, and shows that the error bound of the augmented samples are independent of the size of the neighborhood used for data augmentation.**

I. Background

A. Pre-computed control policies

Implementing optimization-based controllers such as model predictive control (MPC) for fast dynamic systems with limited computing and memory capacity has motivated research on pre-computing and storing the optimal control policy offline such that it can be used online without recomputing the optimization problem. This was first studied under the framework of explicit MPC for linear time-invariant systems [1]. However, the synthesis of the explicit MPC law does not scale well, since the number of piecewise affine polyhedral regions required to capture the MPC control law can increase exponentially with problem size. Extension to nonlinear systems and economic objectives are also not trivial.

An alternative approach is to approximate the optimal policy using parametric function approximators, such as deep neural networks, which are broadly studied under the context of *learning from demonstrations*. The idea of approximating an MPC policy using neural networks was first proposed in [2], but this idea remained more or less dormant (due to the high cost of offline training). However, with the recent promises of deep learning, there has been an unprecedented surge of interest in approximating control policies, see for example recent tutorial paper [3] and the references therein.

Research developments in this direction have been predominantly devoted to understanding the safety and performance of approximate policies [4]–[7]. Although, these are very important developments in the direction of MPC policy approximation, a major challenge of this approach that hinders practical implementation is the cost of training the policy, which is not well studied in the literature, as also noted in [8].

The author is with the Department of Mechanical Engineering, Eindhoven University of Technology, 5600 MB, Eindhoven, The Netherlands. d.krishnamoorthy@tue.nl

B. High cost of offline learning

Consider the "expert" policy which is given by solving the MPC problem

$$V^*(x(t)) = \min_{x_k, u_k} \sum_{k=0}^{N-1} \ell_Q(x_k, u_k) + \ell_P(x_N) \tag{1a}$$

$$\text{s.t. } x_{k+1} = f(x_k, u_k, d) \quad \forall k \in \mathbb{I}_{0:N-1} \tag{1b}$$

$$x \in \mathcal{X}, \quad u \in \mathcal{U}, \quad x_N \in \mathcal{X}_f \tag{1c}$$

$$x_0 = x(t) \tag{1d}$$

where $x \in X \subseteq \mathbb{R}^{n_x}$, $u \in \mathcal{U} \subseteq \mathbb{R}^{n_u}$, and $d \in \mathbb{R}^{n_d}$ denote the states, inputs, and the model parameters, respectively. $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_d} \to \mathbb{R}^{n_x}$ denotes the system model, $\ell_Q : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}$ and $\ell_P : \mathbb{R}^{n_x} \to \mathbb{R}$ denote the stage and terminal cost, which are parameterized by the tuning weights Q and P, respectively, which are treated very generically in this letter to include different MPC formulations, including economic stage and terminal costs. N is the length of the prediction horizon, (1c) denotes the path and terminal constraints, and (1d) denotes the initial condition constraint. Solving (1) at each time step and implementing the first control input gives the implicit MPC policy $u^* = \pi^*(x(t))$.

Approximating the expert policy $\pi^*(x)$ using supervised learning requires a data set consisting of the expert demonstrations in the form of optimal state-action pairs $\mathcal{D} := \{(x_i, u_i^*)\}_{i=1}^{N_s}$, which tells us what action u_i^* the expert (i.e., MPC) would take at state x_i. Using this data set, we can learn a parameterized policy $\pi(x;\theta)$ that tells what actions to take as a function of the current state x, such that $\pi(x;\theta)$ mimics $\pi^*(x)$. The data set $\mathcal{D}$ is generated by solving the MPC problem offline for different realizations of x_i, ideally covering the entire feasible state space $\mathcal{X}_{feas}$.

The availability of a sufficiently rich training data set covering the entire state space is a key stipulation for satisfactory learning [4], [8]. In fact, it has also been shown that the learned policy can result in instability when insufficient demonstrations are used for policy fitting [9], [10]. This means the MPC problem (1) must be solved for several different state realizations x_i, the size of which increases exponentially as the problem dimension increases, (much like the scalability issues with explicit MPC, albeit offline). For example, if we want to include at least s samples along each dimension, then we need to generate at least $N_s = s^{n_x}$ samples. To this end, the first challenge is the cost of generating the data set $\mathcal{D} := \{(x_i, u_i^*)\}_{i=1}^{N_s}$.

Assume now that a sufficiently rich data set $\mathcal{D}$ has been generated offline covering $\mathcal{X}$, and an approximate policy $\pi(x;\theta)$ is learned and has passed the necessary statistical validations and is certified for online use. Now, if any of the MPC tuning parameters Q, P change, or if the model parameter d is updated online (which is not uncommon), then this renders the learned policy $\pi(x;\theta)$ useless. A new training data set $\mathcal{D}$ now has to be generated using the modified MPC formulation and the approximate policy must be re-trained and re-validated from scratch. This is perhaps one of the biggest drawbacks of MPC policy approximation. In theory, this can of course be circumvented by training a parametric function that is also a function of the weights and model parameters $\pi(p;\theta)$, where for example $p := [x, \text{vec}(Q), \text{vec}(P), d]^\mathsf{T}$ and $\pi : \mathcal{P} \rightarrow \mathcal{U}$ with $\mathcal{P}$ being the the combined state and parameter space. However, this makes the offline training problem even more expensive since the dimension of p can be extremely large. Despite the surge of interest in MPC policy approximation, its practical impact will depend on addressing the pivotal challenge of offline data generation.

C. Data Augmentation

In machine learning literature, and more particularly, computer vision and image classification, the issue of data-efficiency is addressed using a simple yet powerful technique known as "Data Augmentation", broadly defined as a *strategy to artificially increase the number of training samples using computationally inexpensive transformations of the existing samples* [11]–[13]. When the data set is a collection of images (i.e. pixel-based data), data augmentation techniques include geometric transformations (such as rotate, translate, crop, flip, etc.), photometric transformations (such as colorize, saturation, contrast, brightness, etc.), and elastic transformation (such as deformation, shear, grid distortion, etc.) of existing images to artificially augment several additional training samples.

Unfortunately, when it comes to data sets consisting of optimal state-action pairs sampled from an optimal policy such as $\pi^*(x)$, the existing data augmentation methods are not applicable. By data augmentation for optimal policy observations, we mean the following: Given an optimal state-action pair (x_i, u_i^*) observed by querying the expert policy $\pi^*(x_i)$, we would like to augment additional data points using computationally inexpensive transformations that would tell us what the optimal action u_j^* would be for states $x_j := x_i + \Delta x$ in the vicinity of x_i. If we can get a reasonable approximation $\hat{u}_j \approx u_j^*$ without actually querying the expert, then the inexact optimal state-action pair $(x_j, \hat{u}_j)$ can be augmented to the set of demonstrations used for learning the policy. In other words, instead of simply learning from a given set of expert demonstrations, we would like to augment additional samples that are *inferred* from the expert demonstrations, thereby enabling us to generalize to states not included in the original set of expert queries.

Noting that the MPC problem (1) is parametric in the initial condition, a sensitivity-based data augmentation scheme amenable for optimal state-action pairs was recently proposed in [14], where it was shown that augmenting additional samples using parametric sensitivities only require the solution to a system of linear equations, which is computationally much cheaper than solving the optimization problem. In [14], the error bound between the expert policy and approximate policy learned from the augmented data set was shown to depend on the size of the region Δx used for data augmentation. Therefore, if we want a certain desired accuracy, depending on the problem Lipschitz properties, this limits the size of Δx that can be used for data augmentation. Building on our recent work [14], the main contribution of this letter paper is an improved data augmentation scheme, where the augmented data samples are enforced to be within an user-specified accuracy by using additional corrector steps. This would make the error bound independent of the size of Δx, thereby enabling us to augment samples from a larger neighborhood Δx without jeopardizing accuracy.

II. Sensitivity-based data augmentation

A. Preliminaries

For the sake of generality, we rewrite the MPC problem (1) in the standard parametric NLP form

$$\Pi(p): \quad \min_{\mathbf{w}} J(\mathbf{w}, p) \tag{2a}$$
$$\text{s.t. } c(\mathbf{w}, p) = 0, \quad g(\mathbf{w}, p) \leq 0 \tag{2b}$$

where $p \in \mathbb{R}^{n_p}$ includes current state of the system x_i, vectorized tuning parameters Q, P, system model parameters d, or any other parameter that can change during online deployment. $\mathbf{w} \in \mathbb{R}^{n_w}$ denotes the primal decision variables (with the optimal action $u_i^* \in \mathbb{R}^{n_u}$ contained within $\mathbf{w}^*$), $J : \mathbb{R}^{n_p} \times \mathbb{R}^{n_w} \rightarrow \mathbb{R}$ denotes the objective function, $c : \mathbb{R}^{n_p} \times \mathbb{R}^{n_w} \rightarrow \mathbb{R}^{n_c}$ and $g : \mathbb{R}^{n_p} \times \mathbb{R}^{n_w} \rightarrow \mathbb{R}^{n_g}$ denotes the set of equality and inequality constraints, respectively.

The Lagrangian of the optimization problem (2) is given by $\mathcal{L}(\mathbf{w}, \lambda, \mu, p) = J(\mathbf{w}, p) + \lambda^\mathsf{T} c(\mathbf{w}, p) + \mu^\mathsf{T} g(\mathbf{w}, p)$ where $\lambda \in \mathbb{R}^{n_c}$ and $\mu \in \mathbb{R}^{n_g}$ are the dual variables. For the inequality constraints, we define $g_{\mathbb{A}} \subseteq g$ as the set of active inequality constraints (i.e. $g_{\mathbb{A}}(\mathbf{w}, p) = 0$) and assume strict complimentarity (i.e., $\mu_{\mathbb{A}} > 0$). The first order necessary conditions of optimality can be denoted compactly as,

$$\varphi(\mathbf{s}(p), p) := \begin{bmatrix} \nabla\mathcal{L}(\mathbf{w}, \lambda, \mu, p) \\ c(\mathbf{w}, p) \\ g_{\mathbb{A}}(\mathbf{w}, p) \end{bmatrix} = 0 \tag{3}$$
$$\mu_{\mathbb{A}} > 0, \mu_{\bar{\mathbb{A}}} = 0, g_{\bar{\mathbb{A}}}(\mathbf{w}, p) < 0$$

Any primal-dual vector $\mathbf{s}^*(p) := [\mathbf{w}^*, \lambda^*, \mu^*]^\mathsf{T}$ is called a KKT-point if it satisfies the first order necessary conditions of optimality (3).

Assumption 1: The cost and constraints $J(\cdot,\cdot)$, $c(\cdot,\cdot)$, and $g(\cdot,\cdot)$ of the NLP problem $\Pi(p)$ are twice continuously differentiable in the neighborhood of the KKT point $\mathbf{s}^*(p)$.

Assumption 2: Linear independence constraint qualification, second order sufficient conditions and strict complementarity holds for $\Pi(p)$.

The above assumption implies that the primal-dual solution vector $\mathbf{s}^*(p_i)$ is a unique local minimizer of $\Pi(p)$. Given Assumptions 1 and 2, it was shown in [15] that for parametric perturbations Δp in the neighborhood of p_i, there exists a unique, continuous, and differentiable vector function $\mathbf{s}(p_i + \Delta p)$ which is a KKT point satisfying LICQ and SOSC for $\Pi(p_i + \Delta p)$, and that the solution vector satisfies $\|\mathbf{s}^*(p_i + \Delta p) - \mathbf{s}^*(p_i)\| \le L_s \|\Delta p\|$, where the notation $\|\cdot\|$ by default denotes Euclidean norm.

Theorem 1 ([14]): Given Assumptions 1 and 2, let u_i^* be the optimal policy obtained by querying $\Pi(p_i)$ for some $p_i \in \mathcal{P}$. Then additional samples $\{(p_i + \Delta p_j, u_i^* + \Delta u_j)\}_j$ in the neighborhood of p_i with the same active constraint set $g_{\mathbb{A}}$ can be augmented to the data set without querying $\Pi(p_i + \Delta p_j)$, where $\Delta u_j \subset \Delta \mathbf{s}_j$ is given by the linear predictor $\Delta \mathbf{s}_j = \mathcal{H}_i \Delta p_j$.

Proof: Taylor series expansion of $\mathbf{s}^*(p)$ around p_i, gives

$$\mathbf{s}^*(p_i + \Delta p_j) = \mathbf{s}^*(p_i) + \frac{\partial \mathbf{s}^*}{\partial p}\Delta p_j + \mathcal{O}(\|\Delta p_j\|^2) \tag{4}$$

Since $\mathbf{s}^*(p)$ satisfies (3) for any p in the neighborhood of p_i, using the implicit function theorem, we have

$$\frac{\partial}{\partial p}\varphi(\mathbf{s}^*(p), p)\bigg|_{p=p_i} = \frac{\partial \varphi}{\partial \mathbf{s}}\frac{\partial \mathbf{s}^*}{\partial p} + \frac{\partial \varphi}{\partial p} = 0$$

$$\Rightarrow \frac{\partial \mathbf{s}^*}{\partial p} = -\left[\frac{\partial \varphi}{\partial \mathbf{s}}\right]^{-1}\frac{\partial \varphi}{\partial p} =: \mathcal{H}_i$$

Substituting this in (4) and ignoring the higher order terms, we get the linear predictor

$$\mathbf{s}^*(p_i + \Delta p_j) \approx \hat{\mathbf{s}}(p_i + \Delta p_j) = \mathbf{s}^*(p_i) + \underbrace{\mathcal{H}_i \Delta p_j}_{:=\Delta \mathbf{s}_j} \tag{5}$$

which contains the inexact optimal action $\hat{u}_j = u_i^* + \Delta u_j$. ■

Remark 1 (Matrix factorization and inverse): The main computation in the linear predictor involves getting $\mathcal{H}_i$. Notice that the first term $\frac{\partial \varphi}{\partial \mathbf{s}}$ is nothing but the KKT matrix, which is already factorized when solving the NLP $\Pi(p_i)$ using e.g. IPOPT [16]. More importantly, augmenting any number of data samples using a single data sample (p_i, u_i^*) requires computing $\mathcal{H}_i$ only once! That is, in the simplest case, using only a single NLP solve at x_i, and a single linear solve (to compute $\mathcal{H}_i$), we can efficiently augment several data samples $\Delta s_j = \mathcal{H}_i \Delta p_j$.

To understand the effect of augmenting the data set with inexact samples, consider the case where the true solution manifold of $\Pi(p)$ is given by $\mathbf{s}^*(p)$ for all $p \in \mathcal{P}$, which is sampled at N_s discrete points $\{p_i\}_{i=1}^{N_s} \in \mathcal{P}$. Assume that there exists a piece-wise affine inexact solution manifold, denoted by $\hat{\mathbf{s}}(p)$ for all $p \in \mathcal{P}$ given by the linear predictor (5) within a neighborhood $\Delta\mathcal{P}_i$ around each p_i for all $i = 1, \dots, N_s$. We make the following assumption regarding the neighborhood $\Delta\mathcal{P}_i$.

Assumption 3: For all $i = 1, \dots, N_s$, the neighborhood $\Delta\mathcal{P}_i$ is chosen around each p_i such that $p_i \in \langle \Delta\mathcal{P}_i \rangle$, $\bigcup_{i=1}^{N_s} \Delta\mathcal{P}_i = \mathcal{P}$, and that the active constraint set remains the same within each neighborhood $\Delta\mathcal{P}_i$, and the solution vector satisfies $\|\mathbf{s}^*(p_i + \Delta p) - \mathbf{s}^*(p_i)\| \le L_s \|\Delta p\|$.

We construct the idea of such a piece-wise affine inexact manifold $\hat{\mathbf{s}}(p)$ given by the linear predictor (5), so that the augmented data points are considered to be sampled from this inexact manifold. Learning the policy then involves fitting a parametric function $\pi(x, \theta)$ to the augmented training data set $\hat{u}(p) \subset \hat{\mathbf{s}}(p)$, such that

$$\theta_1 = \arg\min_{\theta} \sum_{i=1}^{N_s+M} \|\pi(p_i; \theta) - \hat{u}(p_i)\|^2 \tag{6}$$

Assumption 4: The functional form of $\pi(p; \theta)$ has sufficiently rich parametrization and $\exists\ \hat{\theta}$ such that $\hat{u}(p) = \pi(p; \hat{\theta})$.

Theorem 2 ([14]): Consider a problem with the same setup as in Theorem 1, where the base data set $\mathcal{D}^0$ with N_s samples is obtained by querying $\Pi(p)$, which is augmented with M inexact samples from $\hat{u}(p)$ using (5). Under Assumptions 3 and 4,

$$\|\pi(p; \theta_1) - \pi^*(p)\| \le \max\left(\left\{L_{p_i} R_i^2\right\}_{i=1}^{N_s}\right) \tag{7}$$

in probability as $M \to \infty$ holds if θ_1 is a consistent estimator of (6), with $R_i := \sup_{\Delta p \in \Delta\mathcal{P}_i} \|\Delta p\|$.

Proof: Due to the continuity and differentiability of $\mathbf{s}^*(p_i)$ [15], the following holds for all $\Delta p \in \Delta\mathcal{P}_i$,

$$\|\hat{\mathbf{s}}(p_i + \Delta p) - \mathbf{s}^*(p_i + \Delta p)\| \le L_{p_i} \|\Delta p\|^2 \le L_{p_i} R_i^2$$

Aggregating over all the regions results in the inequality

$$\|\hat{\mathbf{s}}(p) - \mathbf{s}^*(p)\| \le \max\left(\left\{L_{p_i} R_i^2\right\}_{i=1}^{N_s}\right) \quad \forall p \in \mathcal{P}$$

Since $u \subset \mathbf{s}(p)$, we have $\|\hat{u}(p) - u^*(p)\| \le \|\hat{\mathbf{s}}(p) - \mathbf{s}^*(p)\|$, which quantifies the maximum deviation between the optimal policy and the augmented inexact samples. If θ_1 is a consistent estimator of (6), then inequality (7) follows under Assumption 4. ■

From this we can see that the error due to learning a policy based on augmented data samples depends on the problem Lipschitz properties L_{p_i}, and the size of the neighborhood Δp_i used to augment the data samples. If one were to use a large neighborhood Δp_i in order to reduce the number of offline NLP computations, then this can potentially affect the performance of the learned policy.

B. Data augmentation with user-enforced accuracy

In order to address this issue, we now propose an improved data augmentation scheme, where in addition to the linear predictor, corrector steps are taken to reduce the approximation error. Adding the KKT conditions to the linear predictor step, and setting $\Delta p = 0$ gives us the SQP correction step

$$\begin{bmatrix} H & \nabla c & \nabla g_{\mathbb{A}} \\ \nabla c^{\mathsf{T}} & 0 & 0 \\ \nabla g_{\mathbb{A}}^{\mathsf{T}} & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta w \\ \Delta\lambda + \lambda^* \\ \Delta\mu_{\mathbb{A}} + \mu_{\mathbb{A}}^* \end{bmatrix} + \begin{bmatrix} \nabla J \\ c \\ g_{\mathbb{A}} \end{bmatrix} = 0 \tag{8}$$

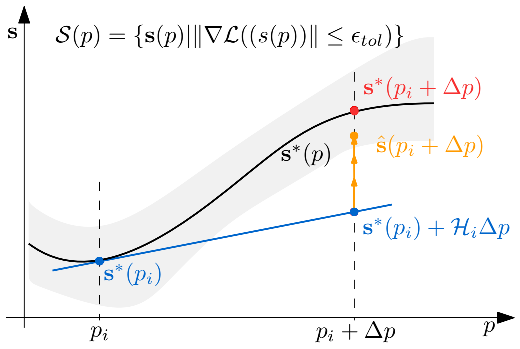
$\mathbf{s}$
$\mathcal{S}(p) = \{\mathbf{s}(p) | \|\nabla\mathcal{L}((s(p))\| \le \epsilon_{tol}\}$
$\mathbf{s}^*(p_i + \Delta p)$
$\hat{\mathbf{s}}(p_i + \Delta p)$
$\mathbf{s}^*(p)$
$\mathbf{s}^*(p_i) + \mathcal{H}_i\Delta p$
$\mathbf{s}^*(p_i)$
p_i
$p_i + \Delta p$
p

Fig. 1: Schematic illustration of the proposed sensitivity-based data augmentation. Blue line indicate the linear predictor step, and the orange arrows indicate the corrector steps. The set $\mathcal{S}(p)$ is shown in gray.

Expanding $\hat{\mathbf{s}}(p_i+\Delta p_j) := [\hat{\mathbf{w}}_{ij}, \hat{\lambda}_{ij}, \hat{\mu}_{ij}]^\mathsf{T}$, the approximate solution is updated as

$$\begin{bmatrix} \hat{\mathbf{w}}_{ij} \\ \hat{\lambda}_{ij} \\ \hat{\mu}_{ij} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{w}}_{ij} \\ 0 \\ 0 \end{bmatrix} - \underbrace{\begin{bmatrix} H & \nabla c & \nabla g_{\mathbb{A}} \\ \nabla c^\mathsf{T} & 0 & 0 \\ \nabla g_{\mathbb{A}}^\mathsf{T} & 0 & 0 \end{bmatrix}^{-1}}_{:=\mathcal{M}_{ij}} \underbrace{\begin{bmatrix} \nabla J \\ c \\ g_{\mathbb{A}} \end{bmatrix}}_{:=\mathcal{Q}_{ij}} \tag{9}$$

where H (Hessian of the Lagrangian), ∇c, $\nabla g_{\mathbb{A}}$,∇J, c, and $g_{\mathbb{A}}$ that make up $\mathcal{M}_{ij}$ and $\mathcal{Q}_{ij}$ are all evaluated at $\hat{\mathbf{s}}(p_i + \Delta p_j)$. The corrector steps are taken, until the optimality residual defined by $\|\nabla\mathcal{L}(\hat{\mathbf{s}}(p_i + \Delta p_j))\|$ is less than some user defined tolerance ϵ_{tol}. If the active constraint set $g_{\mathbb{A}}$ remains the same (cf. Assumption 3), then the corrector step corresponds to a Newton direction on the KKT condition (cf. (8)). For detailed description of the corrector step, see the extended version [17] or [18] and the references therein. The proposed data augmentation scheme with predictor-corrector steps is summarized in Algorithm 1.

Theorem 3: Consider a problem with the same setup as in Theorem 1, where the base data set $\mathcal{D}^0$ with N_s samples is obtained by querying $\Pi(p)$, which is augmented with $M = N_s m$ inexact samples from $\hat{\mathbf{s}}(p)$ using Algorithm 1. Under Assumption 4,

$$\|\pi(p;\theta_1) - \pi^*(p)\| \le r(p) \tag{10}$$

holds in probability as $M \to \infty$ if θ_1 is a consistent estimator of (6), where $r(p) := \sup_{\mathbf{s}(p)\in\mathcal{S}(p)} \|\mathbf{s}(p) - \mathbf{s}^*(p)\|$, with $\mathcal{S}(p) := \{\mathbf{s}(p) | \|\nabla\mathcal{L}(\hat{\mathbf{s}}(p))\| \le \epsilon_{tol}\}$.

Proof: At the solution manifold $\mathbf{s}^*(p)$, we have $\nabla\mathcal{L}(\mathbf{s}^*(p)) = 0$. From the continuity of the cost and constraints, $\exists\ \mathcal{S}(p) := \{\mathbf{s}(p) \mid \|\nabla\mathcal{L}(\mathbf{s}(p))\| \le \epsilon_{tol}\}$ containing the solution manifold $\mathbf{s}^*(p)$ in its interior (cf. Fig. 1) for all p. Solving the corrector steps until the bound on the optimality residual satisfies $\|\nabla\mathcal{L}(\hat{\mathbf{s}}(p))\| \le \epsilon_{tol}$ (cf. line 10 in Algorithm 1) implies that the $\hat{\mathbf{s}}(p) \in \mathcal{S}(p)$. Defining the maximum distance from the exact solution manifold $\mathbf{s}^*(p)$ to the boundary of the set $\mathcal{S}(p)$ for any p as

$$r(p) := \sup_{\mathbf{s}(p)\in\mathcal{S}(p)} \|\mathbf{s}(p) - \mathbf{s}^*(p)\|$$

Algorithm 1 Improved predictor-corrector sensitivity-based data augmentation.

Input: $\Pi(p)$, $\mathcal{P}$, $\mathcal{D} = \emptyset$, ϵ_{tol}

```
1: for $i = 1, \dots, N_s$ do
2:     Sample $p_i \in \mathcal{P}$
3:     $\mathbf{s}^*(p_i) \leftarrow$ Solve $\Pi(p_i)$
4:     Extract $u_i^*$ from the solution vector $\mathbf{s}^*(p_i)$
5:     $\mathcal{D} \leftarrow \mathcal{D} \cup \{(p_i, u_i^*)\}$
6:     $\mathcal{H}_i \leftarrow -\left[\frac{\partial\varphi(\mathbf{s}^*(p_i),p_i)}{\partial\mathbf{s}}\right]^{-1} \frac{\partial\varphi(\mathbf{s}^*(p_i),p_i)}{\partial p}$
7:     for $j = 1, \dots, m$ do
8:         Sample $\Delta p_j \in \Delta\mathcal{P}_i$ in the neighborhood of $p_i$
9:         $\hat{\mathbf{s}}(p_i + \Delta p_j) \leftarrow \mathbf{s}^*(p_i) + \mathcal{H}_i\Delta p_j$    ▷ predictor
10:        while $\|\nabla\mathcal{L}(\hat{\mathbf{s}}(p_i + \Delta p_j))\| > \epsilon_{tol}$ do ▷ corrector
11:            $\hat{\mathbf{s}}(p_i + \Delta p_j) \leftarrow \begin{bmatrix} \hat{\mathbf{w}}_{ij} \\ 0 \\ 0 \end{bmatrix} - \mathcal{M}_{ij}^{-1}\mathcal{Q}_{ij}$
12:        end while
13:        Extract $\hat{u}_j^*$ from the solution vector $\hat{\mathbf{s}}^*(p_i + \Delta p_j)$
14:    end for
15: end for
```

Output: $\mathcal{D}$

results in the inequality $\|\hat{\mathbf{s}}(p) - \mathbf{s}^*(p)\| \le r(p)$. Since $u \subset \mathbf{s}(p)$, we have $\|\hat{u}(p) - u^*(p)\| \le \|\hat{\mathbf{s}}(p) - \mathbf{s}^*(p)\|$. Furthermore, from Assumption 4, we have that

$$\|\hat{u}(p) - u^*(p)\| = \|\pi(p;\hat{\theta}) - \pi^*(p)\| \le r(p)$$

Our result follows, if θ_1 is a consistent estimator of (6). ∎

The bound $r(p)$ clearly depends on ϵ_{tol}, which can be controlled by the user, as opposed to the bound in Theorem 2, which depends on the distance from the original sample.

Remark 2 (Augmented samples): The augmented samples can be approximated from the NLP sample, or another previously augmented sample (similar to a path-following scheme). In either case, corrector steps are taken until the optimality residual is less than ϵ_{tol}. Therefore, Theorem 3 is valid in both cases.

Remark 3 (Active constraint set): Note that if a parturbation in p changes the active constraint set $g_{\mathbb{A}}$, this would induce non-smooth points in the solution manifold $\mathbf{s}^*(p)$. Capturing the non-smooth points in $\mathbf{s}^*(p)$ using piecewise-linear prediction manifolds requires solving a predictor-corrector QP [18], [19]. However, in the context of data augmentation, a simpler alternative could be to discard any samples $p_i + \Delta p$ that induce a change in the active constraint set. In other words, Assumption 3 limits the size of the neighborhood around which additional samples can be augmented.

III. Numerical Experiments

We demonstrate the performance of the data augmentation framework using the benchmark inverted pendulum problem with the nonlinear dynamics $ml^2\ddot{\omega} = u - b\dot{\omega} - mgl\sin\omega$ described by the two states, angle ω, and angular velocity

TABLE I: Comparison of the total CPU time required to generate the different data sets, and the maximum error.

	full NLP	predictor only		predictor-corrector	
	CPU time [s]	CPU time [s]	Max Error	CPU time [s]	Max Error
Case 1	466.42	4.92	1.34	11.95	0.0074
Case 2	456.21	0.45	12.51	8.67	0.028
Case 3	542.73	0.15	79.88	12.96	0.018

$\dot{\omega}$. We consider the case where the expert policy is given by a nonlinear model predictive controller with an objective to drive the pendulum to its inverted position. The input u is the torque, which is used to maintain the pendulum at its inverted position of $\omega = 3.14$ rad, $\dot{\omega} = 0$. In this example, we have $p = [\omega, \dot{\omega}]^{\mathsf{T}}$ and $\mathcal{P} = [0, 2\pi] \times [-5, 5]$. Note that we deliberately choose this simple example to facilitate visualization of the exact and inexact solution manifolds to illustrate the data augmentation schemes.

To approximate the MPC policy, we wish to generate 100×100 state-action data pairs using a grid-based sampling strategy to evenly cover the parameter space $\mathcal{P}$, which would normally take 10^4 offline NLP computations. In each of the following cases, we also solve the full NLP to compute $\pi^*(p)$ at each sample to serve as a benchmark. The NLP problems $\Pi(p)$ are solved using `IPOPT` with `MUMPS` linear solver. The optimization problem and the NLP sensitivities were formulated using `CasADi v3.5.1` [20]. All augmented samples are based on the sensitivity updates from the exact sample obtained by solving the NLP. All computations were performed on a 2.6 GHz processor with 16GB memory.

Case 1: Solve 100 offline NLP problems: In this case, we queried the NMPC expert to generate $N_s = 100$ samples using a 10×10 grid as shown in Fig. 2 in red circles. Using each data sample, we further augment additional 100 data samples around each p_i, leading to a total of $M = 10^4$ augmented data samples using only the linear predictor as done in [14], as well as the proposed improved predictor-corrector data augmentation scheme. These are shown in Fig. 2 in gray dots, where it can be seen that both the data augmentation schemes are able to sufficiently capture the solution manifold. The maximum error due to approximation when using only the linear predictor is 1.34, whereas the maximum error was 0.0074 with the improved data augmentation.

Case 2: Solve 9 offline NLP problems: We then consider the case where we queried the NMPC expert only $N_s = 9$ times using a 3×3 grid as shown in Fig. 2 in red circles. Using each data sample, we further augment additional 1089 data samples around each sample, leading to a total of 9081 augmented data samples using only 9 offline NLP computations. Since the size of the neighborhood increases, the inexact solution manifold using only the linear predictor is now visbily different from the exact solution manifold (with 9 piecewise affine manifolds), with the maximum error of 12.5. On the other hand, when using the improved data augmentation scheme, the maximum error is only 0.028. This shows that despite the large neighborhood used for data augmentation, we can get the generate data samples with a desired accuracy at only marginally increased computational cost (cf. Table I).

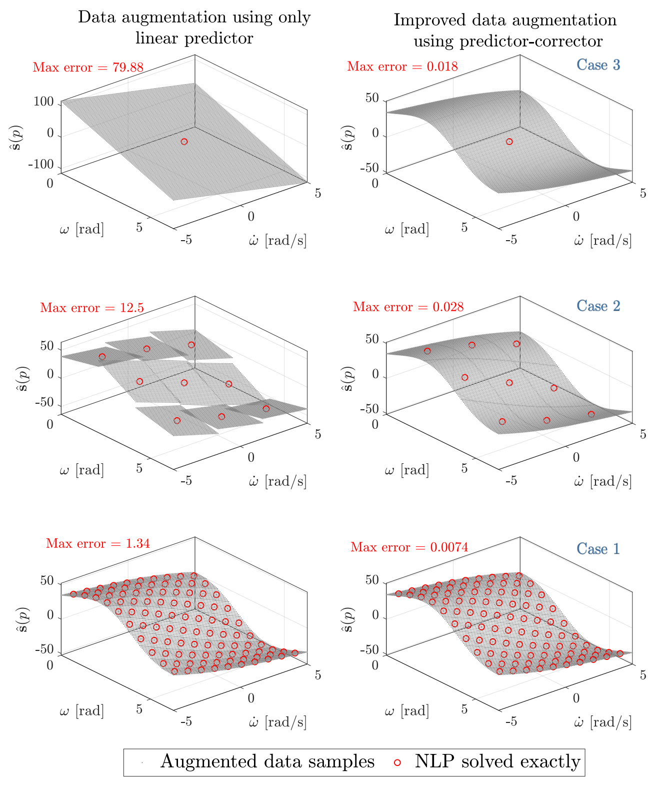

Fig. 2: Data set with optimal state-action data pairs for the inverted pendulum example. Red circles indicate samples where the MPC problem is solved exactly, and gray dots indicate samples that are augmented using only the linear predictor [14] (left subplots), and the proposed approach using additional corrector steps (right subplots).

Case 3: Solve only 1 offline NLP problem: We then consider the extreme case, where we only solve $N_s = 1$ offline NLP problem as shown in Fig. 2. Using this single NLP computation, we augment 10000 data samples using a 100×100 grid. Using only the linear predictor, naturally we get a single affine manifold, leading to large approximation errors. However, using the improved data augmentation scheme, we are still able to augment data samples with a desired accuracy, where the maximum error is only 0.018, as opposed to 79.88.

The total CPU time to generate the data samples using the two data augmentation schemes, as well as solving all the data points exactly along with the maximum error are summarized in Table I, which also shows the trade-off between computation cost and accuracy. Since the main focus of this letter is on data generation, and not on policy approximation itself, closed-loop simulation results using the learned policy for the different cases is not shown here. However, the interested reader can find this in the extended version [17].

IV. Discussion

The challenge of generating large data sets, if at all considered in the current works, is mainly addressed via efficient sampling techniques. This warrants a discussion on how our proposed data augmentation scheme fits in the context of the different sampling strategies that have been proposed in the literature. A recursive sampling algorithm was proposed in [21], where new sample locations are chosen with higher resolution as long as neighboring values are still differing. The authors in [8] recently proposed an algorithm to efficiently generate large data sets based on geometric random walks instead of independent sampling, where starting from a feasible state, small steps are taken iteratively along a random line until each step is feasible. A control oriented sampling strategy was proposed in [22] based on the premise that it is unlikely that all states from the feasible region are equally likely to be observed during closed-loop operation. In this approach, starting from a feasible initial state, the consecutive samples are selected based on noisy closed-loop simulations, such that the samples are selected from a tube of closed loop trajectories likely to be observed in operation. All these sampling strategies help us decide where to sample the state-space, and at each sample location, the optimization problem is solved exactly to get the corresponding optimal action. As such all these algorithm naturally lend itself to our proposed data augmentation scheme. For example, if a sample location determined using any sampling strategy [8], [21], [22] is within a neighborhood of a previously computed sample satisfying the assumptions of Theorem 1, then one can use the proposed data augmentation instead of solving the MPC problem exactly. Thus the proposed method complements the different sampling strategies [8], [21], [22].

V. Conclusion

To summarize, this paper presented an improved data augmentation scheme for cheaply generating the training data samples for MPC policy approximation using only a fraction of the computation effort. We showed that by using additional corrector steps we can enforce a desired level of accuracy in the augmented samples (cf. Algorithm 1), which amounts to a few additional linear solves. By doing so the approximation error does not depend on the size of the neighborhood used for data augmentation, but on the user-defined optimality residual (cf. Theorem 3), which allows us to augment samples from larger neighborhoods while without jeopardizing accuracy. This was demonstrated on a inverted pendulum control problem, which clearly shows the trade-off between accuracy and computational effort. The proposed data augmentation approach can be used with any sampling strategy, as well as for a broad class of *learning from demonstration* problems including imitation learning with interactive expert, value function approximation, and inverse optimal control.

References

[1] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.

[2] T. Parisini and R. Zoppoli, "A receding-horizon regulator for nonlinear systems and a neural approximation," *Automatica*, vol. 31, no. 10, pp. 1443–1451, 1995.

[3] A. Mesbah, K. P. Wabersich, A. P. Schoellig, M. N. Zeilinger, S. Lucia, T. A. Badgwell, and J. A. Paulson, "Fusion of machine learning and mpc under uncertainty: What advances are on the horizon?" in *2022 American Control Conference (ACC)*. IEEE, 2022, pp. 342–357.

[4] M. Hertneck, J. Köhler, S. Trimpe, and F. Allgöwer, "Learning an approximate model predictive controller with guarantees," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 543–548, 2018.

[5] X. Zhang, M. Bujarbaruah, and F. Borrelli, "Near-optimal rapid mpc using neural networks: A primal-dual policy learning framework," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 5, pp. 2102–2114, 2020.

[6] J. A. Paulson and A. Mesbah, "Approximate closed-loop robust model predictive control with guaranteed stability and constraint satisfaction," *IEEE Control Systems Letters*, 2020.

[7] B. Karg, T. Alamo, and S. Lucia, "Probabilistic performance validation of deep learning-based robust NMPC controllers," *International Journal of Robust and Nonlinear Control*, vol. 31, no. 18, pp. 8855–8876, 2021.

[8] S. W. Chen, T. Wang, N. Atanasov, V. Kumar, and M. Morari, "Large scale model predictive control with neural networks and primal active sets," *Automatica*, vol. 135, p. 109947, 2022.

[9] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.

[10] M. Palan, S. Barratt, A. McCauley, D. Sadigh, V. Sindhwani, and S. Boyd, "Fitting a linear control policy to demonstrations with a kalman constraint," in *Learning for Dynamics and Control*. PMLR, 2020, pp. 374–383.

[11] T. Tran, T. Pham, G. Carneiro, L. Palmer, and I. Reid, "A bayesian data augmentation approach for learning deep models," in *Advances in neural information processing systems*, 2017, pp. 2797–2806.

[12] L. Taylor and G. Nitschke, "Improving deep learning with generic data augmentation," in *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2018, pp. 1542–1547.

[13] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, pp. 1–48, 2019.

[14] D. Krishnamoorthy, "A sensitivity-based data augmentation framework for model predictive control policy approximation," *IEEE Transactions on Automatic Control*, vol. In-Press, 2021.

[15] A. V. Fiacco, "Sensitivity analysis for nonlinear programming using penalty methods," *Mathematical programming*, vol. 10, no. 1, pp. 287–311, 1976.

[16] H. Pirnay, R. López-Negrete, and L. T. Biegler, "Optimal sensitivity based on IPOPT," *Mathematical Programming Computation*, vol. 4, no. 4, pp. 307–331, 2012.

[17] D. Krishnamoorthy, "An improved data augmentation scheme for model predictive control policy approximation," *arXiv:2303.05607 [eess.SY]*, 2023.

[18] V. Kungurtsev and J. Jäschke, "A predictor-corrector path-following algorithm for dual-degenerate parametric optimization problems," *SIAM Journal on Optimization*, vol. 27, no. 1, pp. 538–564, 2017.

[19] J. F. Bonnans and A. Shapiro, "Optimization problems with perturbations: A guided tour," *SIAM review*, vol. 40, no. 2, pp. 228–264, 1998.

[20] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[21] J. Nubert, "Learning-based approximate model predictive control with guarantees - joining neural networks with recent robust MPC," Master's thesis, ETH Zürich, 2019.

[22] D. Krishnamoorthy, A. Mesbah, and J. A. Paulson, "An adaptive correction scheme for offset-free asymptotic performance in deep learning-based economic MPC," *IFAC-PapersOnLine*, vol. 54, no. 3, pp. 584–589, 2021.