Engineering Applications of Artificial Intelligence 77 (2019) 1–8

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Machine learning-based warm starting of active set methods in embedded model predictive control

Martin Klaučo *, Martin Kalúz, Michal Kvasnica

Slovak University of Technology in Bratislava, Radlinského 9, SK-812 37 Bratislava, Slovak Republic

ARTICLE INFO

Keywords:
Artificial intelligence
Active set method
Model predictive control

ABSTRACT

We propose to apply artificial intelligence approaches in a warm-starting procedure to accelerate active set methods that are used to solve strictly convex quadratic programs in the context of embedded model predictive control (MPC). The proposed warm-starting is based on machine learning where a good initialization of the active set method is learned from training data. Two approaches to generate the training data set are discussed, one based on gridding the feasibility domain, and one based on closed-loop simulations with typical initial conditions. The training data are then processed by machine learning-based classification algorithms that yield a good estimate of the initial active set for the iterative active set algorithm. By means of extensive case studies we demonstrate that the proposed approach is superior to existing warm-starting procedures in that it considerably reduces the number of active set iterations, thus allowing embedded MPC to be implemented using less computational effort.

1. Introduction

Model Predictive Control (MPC) is nowadays a de facto standard control methodology when controlling multivariable systems where satisfaction of constraints and optimization of the control performance are required (Maciejowski, 2002; Rossiter, 2003; Camacho and Bordons, 2007). Most of MPC in practice is formulated and solved as convex quadratic programs (QP), parameterized in the initial state measurement. When controlling systems with fast dynamics, it is vital to be able to solve such QPs fast enough. This, however, becomes challenging when constraints of the implementation hardware, such as the available computational power and memory storage, are taken into account, especially when targeting simple hardware such as microcontrollers (Zometa et al., 2013), field programmable gate arrays (Ling et al., 2008), or programmable logic controllers (Huyck et al., 2012).

Various avenues to providing the solution to a given MPC-based QP with minimal resources (computation and memory) can be pursued. One option is to solve the QP *off-line* for all possible values of the initial condition using parametric programming (Bemporad et al., 2002). Such techniques, however, are limited to problems of small size, say, for a short prediction horizon, up to 10 states and up to 4 control inputs. For problems of bigger size, one usually solves the QP *on-line*. To do so, a plethora of methods have been suggested, such as active set methods (Nocedal and Wright, 2006; Wills and Ninness, 2010; Ferreau et al., 2008), interior point approaches (Rao et al., 1998), fast gradient procedures (Richter et al., 2012; Kögel and Findeisen, 2011), gradient projection algorithms (Patrinos and Bemporad, 2014), splitting methods (O'Donoghue et al., 2013; Stathopoulos et al., 2014), and tailored MPC algorithms (Liu and Kong, 2014), to name just a few. Among these, the active set methods are frequently used due to their simplicity and easiness of implementation in the embedded framework (Cimini and Bemporad, 2017).

In this paper, we focus on active set methods (ASM). There, the working active set $\mathcal{A}$ and the primal/dual optimizers are iteratively updated until the global optimum of the convex QP is found. The dominant speed factors of ASM are the number of iterations and the cost of linear algebra in each iteration, with the former having larger impact on the overall performance (Herceg et al., 2015).

As shown in Herceg et al. (2015), the number of ASM iterations can be significantly decreased if the iterations are *warm started* from some known initial active set $\mathcal{A}_0$ and the associated primal/dual feasible solution. In Ferreau et al. (2008, 2014), the ASM is initialized from the active set $\mathcal{A}^\star(t-1)$ obtained at the previous time instant $t-1$. This, however, does not take into account the current state measurement and therefore works well only when the active sets do not change much in time. An improved warm starting procedure, which takes into account the information about the current state $x(t)$, was proposed in

* Corresponding author.
E-mail addresses: martin.klauco@stuba.sk (M. Klaučo), martin.kaluz@stuba.sk (M. Kalúz), michal.kvasnica@stuba.sk (M. Kvasnica).

https://doi.org/10.1016/j.engappai.2018.09.014
Received 2 March 2018; Received in revised form 13 July 2018; Accepted 18 September 2018

Otta et al. (2015). There, LQR-based warm starting is used in conjunction with projections onto constraints. Therefore the approach is only suitable, from a practical point of view, if the constraint set is simple, such as a hyperbox. Finally, in Zeilinger et al. (2011) the authors have proposed to devise a state-dependent warm start $\mathcal{A}_0(x(t))$ by solving, off-line, a parametric program. Although substantial reduction in the number of iterations could be achieved, the procedure is limited to systems with a modest number of states, say, below ten.

In this paper we propose to apply machine learning (ML) to accelerate primal active set methods used to solve QP-based MPC problems. Specifically, we show how to construct a state-dependent warm start procedure that yields the initial active set $\mathcal{A}_0$ as a function of the current state $x(t)$. Specifically, the policy $\mathcal{A}_0(x(t))$ is learned from training data collected off-line. We show that by resorting to standard ML-based classification algorithms, such as classification trees (Breiman et al., 1984) and k-nearest neighbors (Dasarathy, 1991), the learned policy is simple enough as to enable a fast and cheap embedded implementation, and performs better compared to the feedback warm-starting of Ferreau et al. (2008). Finally, the procedure is applicable to QP-based MPC problems where the constraint set is an arbitrary polyhedron, thus allowing, among others, to include terminal set constraints. In fact, application of machine learning procedures to solve optimization problems is not new. Research presented in Nazemi (2014) and Elsayed et al. (2014) use the neural networks to find optimal solution to the strictly convex QPs.

Note that the proposed machine-learning based acceleration of active set methods *does not replace* the control algorithm, it merely complements it. Specifically, as shown in Section 5, ML allows the ASM to arrive at the optimal solution using a fewer number of iterations compared to conventional methods, such as the one of Ferreau et al. (2008). Since the primal active set method guarantees that the optimal solution is found even if the initialization is not correct, the proposed procedure maintains guarantees of closed-loop stability and constraint satisfaction.

2. Preliminaries

2.1. QP-based MPC

For linear systems

$$x(t+1) = Ax(t) + Bu(t), \tag{1}$$

subject to constraints $x \in \mathcal{X}$, $u \in \mathcal{U}$ with $\mathcal{X} \subseteq \mathbb{R}^n$, $\mathcal{U} \subseteq \mathbb{R}^m$, the context of this paper revolves around solving the following MPC problem:

$$\min_{u_0,\dots,u_{N-1}} \; x_N^\intercal Q_\mathrm{f} x_N + \sum_{k=0}^{N-1} \left(x_k^\intercal Q_\mathrm{x} x_k + u_k^\intercal Q_\mathrm{u} u_k\right) \tag{2a}$$

$$\text{s.t. } x_{k+1} = Ax_k + Bu_k, \quad k = 0,\dots,N-1, \tag{2b}$$

$$(x_k, u_k) \in \mathcal{X} \times \mathcal{U}, \qquad k = 0,\dots,N-1, \tag{2c}$$

$$x_N \in \mathcal{X}_\mathrm{f}, \tag{2d}$$

where $Q_\mathrm{f} \succeq 0$, $Q_\mathrm{x} \succeq 0$, $Q_\mathrm{u} \succ 0$ are penalty matrices, $N \in \mathbb{N} < \infty$ is a finite prediction horizon, A, B are system matrices, and $\mathcal{X}$, $\mathcal{U}$, $\mathcal{X}_\mathrm{f}$ are polyhedra that contain the origin in their respective interiors, and $\mathcal{X}_\mathrm{f} \subset \mathcal{X}$. It is well known, see, e.g., Mayne et al. (2000) that (2) can be converted into a parametric quadratic program

$$\min_U \; 1/2\, U^\intercal H U + \theta^\intercal F U \tag{3a}$$

$$\text{s.t. } GU \leq E\theta + w, \tag{3b}$$

with $\theta = x_0$ being the parameter and $U = [u_0^\intercal, \dots, u_{N-1}^\intercal]^\intercal$ representing the optimization variables. The vectors/matrices H, F, G, E, w can be obtained from (2) using straightforward algebraic manipulations. We remark that $H \succ 0$, thus the Hessian is invertible, since $Q_\mathrm{f} \succeq 0$, $Q_\mathrm{x} \succeq 0$, $Q_\mathrm{u} \succ 0$ is assumed. The optimal open-loop sequence of control actions, i.e., $U^\star$, can then be obtained by solving (3) for a particular initial condition $\theta = x(t)$. MPC is then traditionally implemented in the receding horizon fashion where only the first element of $U^\star$, i.e., $u_0^\star$ is implemented to the controlled system and the whole procedure is repeated at the subsequent time instant for a new value of the initial condition $\theta = x(t)$.

2.2. Primal active set method for strictly convex QPs

The primal active set method (ASM) (Fletcher, 2013) is an iterative procedure for finding the minimizer $U^\star$ to (3) for a fixed value of the parameter θ. At each iteration, the ASM searches for a vector Δ along which the objective function (3a) decreases. To tackle constraints, the procedure also iteratively updates the index set $\mathcal{A}$ of constraints that are active at the current iterate. Specifically, at each iteration the active set method solves the equality-constrained QP (EQP) of the form

$$\min_\Delta \; \frac{1}{2}(U+\Delta)^\intercal H (U+\Delta) + \theta^\intercal F (U+\Delta) \tag{4a}$$

$$\text{s.t. } G_\mathcal{A}(U+\Delta) = E_\mathcal{A}\theta + w_\mathcal{A}, \tag{4b}$$

where $G_\mathcal{A}$ consists of the rows of G indexed by the set $\mathcal{A} \subseteq \{1,\dots,c\}$ where c is the number of constraints in (3b). In (4), U is considered fixed and feasible, i.e., $G_\mathcal{A}U = E_\mathcal{A}\theta + w_\mathcal{A}$, and therefore $G_\mathcal{A}\Delta = 0$ needs to hold to retain feasibility. The improving direction Δ for (4) can be solved from the Karush–Kuhn–Tucker system

$$\begin{bmatrix} \Delta \\ \lambda \end{bmatrix} = \begin{bmatrix} H & G_\mathcal{A}^\intercal \\ G_\mathcal{A} & 0 \end{bmatrix}^{-1} \begin{bmatrix} -HU - F^\intercal\theta \\ 0 \end{bmatrix}, \tag{5}$$

where λ is the vector of Lagrange multipliers. If[1] $\|\Delta\| = 0$ and $\lambda_i \geq 0$ for all $i \in \mathcal{A}$, U is the optimal solution and the iterations terminate. If, on the other hand, some Lagrange multipliers are negative, one constraint is removed from $\mathcal{A}$, typically the one corresponding to the smallest Lagrange multiplier, i.e., $\mathcal{A} = \mathcal{A} \setminus \{i^\star\}$ with

$$i^\star = \arg\min_{i \in \mathcal{A}} \lambda_i, \tag{6}$$

and the procedure continues with the next iteration. If the improving direction Δ is non-zero, the current iterate U is refined by $U = U + \alpha\Delta$. Here, the step size α is given by $\alpha = \min\{1, \beta_j\}$, where β_j are defined, for all $j \notin \mathcal{A}$ for which $G_j\Delta > 0$, as

$$\beta_j = \frac{E_j\theta + w_j - G_j U}{G_j \Delta}, \tag{7}$$

where G_j is the jth row of the matrix. If $\alpha < 1$, then some previously inactive constraint becomes active when updating U along the direction Δ. Among all such constraints, one typically (Fletcher, 2013) picks the one that is activated first, i.e., $j^\star = \arg\min_j \beta_j$, followed by adding $j^\star$ to the active set, i.e., $\mathcal{A} = \mathcal{A} \cup \{j^\star\}$. The complete procedure is reported as Algorithm 1.

Remark 2.1. Algorithm 1 takes the initial active set $\mathcal{A}_0$ and the initial primal solution U_0 as its inputs. If no prior information is available, one can always choose $\mathcal{A}_0 = \emptyset$ and $U_0 = 0$, provided $0 \in \mathcal{U}$. If $\mathcal{A}_0$ is given as a non-empty set, U_0 can be computed by $U_0 = G_{\mathcal{A}_0}^\dagger(E_{\mathcal{A}_0}\theta + w_{\mathcal{A}_0})$ where $\dagger$ denotes the left Moore–Penrose inverse. □

Remark 2.2. As pointed out in Fletcher (2013, Chapter 10.3), Algorithm 1 converges to the primal optimal solution $U^\star$ regardless of the choice of the initial active set $\mathcal{A}_0$, provided it is chosen such that the matrix $G_{\mathcal{A}_0}$ is of full row rank. Naturally, each choice of $\mathcal{A}_0$ leads to a different sequence of iterations. □

3. Problem statement

The number of iterations the ASM takes to converge to the optimal solution $U^\star(t)$ for a given initial condition $\theta = x(t)$ at time instant t

[1] In the floating point environment, it is advised to replace the condition $\|\Delta\| = 0$ in Algorithm 1 by $\|\Delta\| \leq \epsilon$ for some small positive value of ϵ, typically equal to the machine precision.