



Complexity Reduction of Explicit Model Predictive Control via Combining Separator Function and Binary Search Trees

Jamal Arezoo, Karim Salahshoor*

Department of Automation and Instrumentation, Petroleum University of Technology, Ahvaz, Iran

Email address:

jamalarezoo@gmail.com (J. Arezoo), salahshoor@put.ac.ir (K. Salahshoor)

*Corresponding author

To cite this article:

Jamal Arezoo, Karim Salahshoor. Complexity Reduction of Explicit Model Predictive Control via Combining Separator Function and Binary Search Trees. *American Journal of Computer Science and Technology*. Vol. 1, No. 1, 2018, pp. 19-23. doi: 10.11648/j.ajcst.20180101.13

Received: November 21, 2017; **Accepted:** November 29, 2017; **Published:** December 24, 2017

Abstract: The explicit Model Predictive Control (MPC) has emerged as a powerful technique to solve the optimization problem offline for embedded applications where computations is performed online. Despite practical obstacles in implementation of explicit model predictive control (MPC), the main drawbacks of MPC, namely the need to solve a mathematical program on line to compute the control action are removed. This paper addresses complexity of explicit model predictive control (MPC) in terms of online evaluation and memory requirement. Complexity reduction approaches for explicit MPC has recently been emerged as techniques to enhance applicability of MPC. Individual deployment of the approaches has not had enough effect on complexity reduction. In this paper, merging the approaches based on complexity reduction is addressed. The binary search tree and complexity reduction via separation are efficient methods which can be confined to small problems, but merging them can result in significant effect and expansion of its applicability. The simulation tests show proposed approach significantly outperforms previous methods.

Keywords: Multi-Parametric Programming, Saturated Regions, Search Tree, Predictive Control

1. Introduction

Model predictive control is an optimization-based control policy which is able to handle constrained multivariable systems. The ability to solve the mp-QP problem in embedded applications often poses a crucial issue where sampling period is small and storage limitation exists. In the last decade embedded Model Predictive Control (MPC) has become increasingly important in many engineering fields, like automotive, aerospace and power systems. MPC is the accepted control technique for constrained multi-variable systems which are controlled in the receding horizon strategy [1]. Implementation of a typical MPC approach requires to online solve an optimization problem based on system model at each sampling time. A constrained finite-horizon optimal control problem must be solved at each time step, making the online implementation of MPC in embedded boards a challenge [2]. One way to avoid the online solution is to represent the solution explicitly via reformulation of the MPC problem into an equivalent multi-parametric quadratic programming (mp-QP) problem. Traditionally, the optimal

solution is obtained either by an iterative numerical procedure (referred to as implicit MPC), or by evaluating the explicit representation of the MPC feedback law, which is obtained off-line using parametric programming (referred to as explicit MPC). Both approaches have their pros and cons. Implicit MPC requires more computational resources, but is able to handle large systems. Explicit MPC, on the other hand, requires less online computation, but the off-line construction of the feedback law scales badly with increasing dimensionality of the problem. Moreover, the memory footprint of the explicit solutions can easily violate limits of the available memory storage [3].

Explicit MPC drastically reduces the computational load by pre-solving off-line the QP and converting the MPC law into a continuous and piecewise affine (PWA) function of the parameter vector. When an explicit MPC controller is executed one needs to identify which polyhedron region the measured state lies in, and then compute the control actions using the corresponding affine control law [4]. Although successfully applied in several practical applications, especially to automotive systems [5] and power converters [6], explicit MPC tends to generate a large set of controller

gains. The number of gains depends roughly exponentially on the number of constraints included in the MPC optimization problem. To simplify the complexity of explicit MPC controllers, approximate explicit MPC techniques were addressed recently [7] to [9]. In [10], authors adopted a special class of basic functions, the canonical piecewise-affine (PWA) functions described in [11], to approximate a given linear MPC controller and impose constraints in the approximation procedure so as to be able to analyze closed-loop stability properties using PWA Lyapunov functions. In [12] and [13], complexity reduction is carried out via replacing original function with a simpler equivalent function that the replacement result in removing saturated regions and the corresponding optimal control actions attaining a saturated value. In [14] another efficient approach was presented which is based on a data structure for representation of PWA functions to minimize the time needed to evaluate the function. It also reduces storage requirements by this data structure. This approach has offered significant computational advantages [15] because evaluation time is proportional to logarithm of the number regions in the PWA functions that it allows faster evaluation and subsequently real implementation. However, the approach can be confined to small problems.

In this paper, a novel unified method is proposed that it is based on merging the two approaches [14] and [13]. Individual deployment of the mentioned approaches can be prohibitive for implementation of large problems, then to get rid of the obstacle, unifying the approaches is proposed. The idea is composed of two steps. The first step is to removing saturated regions via separator function. The second one is to represent the unsaturated regions and corresponding control actions using search binary tree. The extensive problems demonstrate Merits of the proposed approach especially in implementation.

2. Explicit MPC Problem Statement

Consider the linear time-invariant system model

$$\begin{cases} x_{k+1} = Ax_k + Bu_k \\ y_k = Cx_k \end{cases} \quad (1)$$

Subject to the constraints

$$\begin{cases} x_{k+1} = Ax_k + Bu_k \\ x_{min} \leq x_k \leq x_{max} \\ u_{min} \leq u_k \leq u_{max} \\ y_{min} \leq y_k \leq y_{max} \end{cases} \quad (2)$$

in which $x_k \in R^n$, $u_k \in R^m$ and $y_k \in R^n$ are the state, input and output vector respectively. $x_{min} \leq x_{max}$, $u_{min} \leq u_{max}$ and $y_{min} \leq y_{max}$ are bounds on states, inputs and outputs respectively.

In the following, we concentrate on the constrained finite time optimal control problem of the form:

$$J(x) = \min_U \frac{1}{2} x_N^T P x_N + \frac{1}{2} \sum_{k=0}^{N-1} x_k^T Q x_k + u_k^T R u_k \quad (3)$$

Subject to

$$\begin{cases} x_{k+1} = Ax_k + Bu_k \\ x_{min} \leq x_k \leq x_{max} \\ u_{min} \leq u_k \leq u_{max} \\ y_{min} \leq y_k \leq y_{max} \end{cases} \quad (4)$$

Where $Q \geq 0$, $R > 0$ and $P \geq 0$ are penalty matrices such that the problem (3) is strictly convex. The optimization problem (3) can be rewritten as the QP problem of the form:

$$J(x) = \frac{1}{2} x^T Y x + \min_U \frac{1}{2} U^T H U + x^T F U \quad (5a)$$

Subject to

$$GU \leq W + Ex \quad (5b)$$

where $U = \{u_k, \dots, u_{k+N-1}\} \in R^{m \times N}$ denote the vector of optimization problem and $H \in R^{m \times m}$, $G \in R^{q \times m}$, $W \in R^q$ and $E \in R^{q \times n}$ are real matrices which can be obtained from Q , R and P . We assume that (5b) does not contain any redundant constraints [14].

Theorem 1 [16]. Consider the multi-parametric quadratic program (5) and let $H > 0$. Then the set of feasible parameters X is convex, the optimizer $U(x): X \rightarrow R^{m \times N}$ is continuous and piecewise affine, and the optimal solution $J(x)^*: X \rightarrow R$ is convex and piecewise quadratic. In [16] it was shown that the optimal state-feedback piecewise affine MPC control law can be represented explicitly as follows:

$$u(x) = [\mathbb{I} \ 0 \ \dots \ 0] U = \begin{cases} F_1 x + G_1, & \text{if } x \in R_1 \\ \vdots \\ F_{n_r} x + G_{n_r}, & \text{if } x \in R_{n_r} \end{cases} \quad (6)$$

where $F_i \in R^{m \times n}$ and $G_i \in R^m$ are, respectively, real matrices and vectors of the MPC controller. R_i , $i = 1, \dots, n_r$ are critical regions, and n_r denotes the total number of critical regions.

In [17] and [18], straightforward implementation of piecewise affine function (6) in the order $1\mu s$ was reported. Objective of this paper is to find an alternative representation for (6) so that complexity of the implementation i.e. memory requirement and computational time, can be further reduced.

3. Complexity Reduction via Elimination of Saturated Regions

Partition of regions $\{R_1, \dots, R_{n_r}\}$ can be classified into two types of regions. First of the partition is the regions which their associated control actions can be saturated at the maximal or minimal values i.e. respectively u_{max} or u_{min} . The remains of regions $\{R_1, \dots, R_{n_r}\}$ are called unsaturated regions. With this classification, (6) can be rewritten as follows:

$$u(x) = \begin{cases} F_i x + G_i, & \text{if } x \in R_{unsat} \\ u_{max}, & \text{if } x \in \bar{R}_{sat} \\ u_{min}, & \text{if } x \in \underline{R}_{sat} \end{cases} \quad (7)$$

Where \bar{R}_{sat} and \underline{R}_{sat} denote saturated regions and R_{unsat} denotes unsaturated regions. In the section, the main challenge is to find a separator to make regions \bar{R}_{sat} and \underline{R}_{sat} distinct such that the separator consumes less memory and computation effort. In [13], authors have shown that there is such as separator function which separates the regions of \bar{R}_{sat} and \underline{R}_{sat} as:

$$u(x) = \begin{cases} F_i x + G_i, & \text{if } x \in R_{unsat} \\ u_{max}, & \text{if } x \notin R_{unsat}, \xi(x) > 0, \\ u_{min}, & \text{if } x \notin R_{unsat}, \xi(x) < 0. \end{cases} \quad (8)$$

Indeed, the separation is performed based on sign of the separator function $\xi(x): R^n \rightarrow R$ such that if the sign is positive and the measured state does not lie in the unsaturated regions R_{unsat} then the measured state lies in the regions which their associated control actions are u_{max} . Otherwise, associated control actions are u_{min} . Note that first searching through regions R_{unsat} is performed and then sign of separator function is checked, provided that measured state does not lie in R_{unsat} .

In [13], a polynomial function is introduced as separator function that its coefficients are determined such that it makes the regions \bar{R}_{sat} and \underline{R}_{sat} distinct.

Therefore, instead of evaluation of the saturated regions, only sign of the separator is determined. Then, the saturated regions and their associated control actions are removed that result in complexity reduction in computation time and memory requirement. However, the success of this approach depends mainly on two factors: 1) tightness of input constraints 2) selection of the lower input penalty R in (3) and (4) that leads to the fact that the approach is not general.

4. Complexity Reduction via Binary Search Tree

The approach proposed in previous section can remove the saturated regions which the number of them is problem-dependent and maybe is not large. Then, it always cannot have a large effect on complexity reduction all the time. On the other hand, the number of unsaturated regions is also large.

To increase complexity reduction, the other efficient methods can be employed. This section addresses the approach search tree and its impact on complexity reduction.

The evaluation of PWA functions plays an important role in implementation of explicit MPC. In [14] a binary search tree is constructed on the basis of the hyper planes defining the convex polyhedral regions. The main idea is to evaluate less number of the hyper planes that result in reduction of 1) computation time and 2) storage requirement. This approach results in an efficient online evaluation complexity which is logarithmic in the number of regions (n_r) in the best case, with a balanced search tree. This approach results in an efficient online evaluation complexity which is logarithmic in the number of regions (n_r) in the best case, with a balanced search tree.

Consider the set of polyhedral regions $\{R_1, \dots, R_{n_r}\}$ and the

corresponding set of distinct control action functions $\{f_1, \dots, f_k\}$ where $k \leq n_r$, since some regions may share the same function. Let $\{d_j(x) = a_j^T x - b_j = 0, j = 1, \dots, L\}$ denotes the set of all separating hyper planes defining the polyhedral partition. Define J as the index representation of a polyhedron [14] which consists of the corresponding hyper planes determined by $d_j(x)$ and their sign, e.g., $J = \{1^+, 4^+, 6^-\}$ would mean that $d_1(x) \geq 0$, $d_4(x) \geq 0$, $d_6(x) \leq 0$. Such an index representation obviously defines a polyhedron in the state space i.e., $P(J) = \{x | d_1(x) \geq 0, d_4(x) \geq 0, d_6(x) \leq 0\}$. Also define the index sets $I(J) = \{i | R_i \cap P(J) \text{ is full-dimensional}\}$ and $F(I) = \{K | f_k \text{ corresponds to } R_i, i \in I\}$, where $F(I)$ denotes the index set of all functions corresponding to the index set $I(J)$. See Figure 1 for an example of a simple search tree.

Lemma 1 [14]: if $i \in I(J) \cap I(j^+)$ and $i \notin I(J \cup j^+)$, then R_i is split into two full-dimensional polyhedrons by hyper plane j i.e., $i \in I(J) \cap I(j^+) \cap I(j^-)$. The same result holds when j^+ is interchanged by j^- .

The important advantage of lemma 1 is that $I(J \cup j^+)$ is efficiently approximated by $I(J) \cap I(j^+)$ computationally.

This approach leads to an efficient online evaluation complexity which is logarithmic in the number of regions (n_r) in the best case, with a balanced search tree. These hyper planes is used as separating variables in the nodes of the tree to split the partitioned space in a way that the number of candidate affine functions decreases as much as possible from the current to the next level of the tree [19].



actions. Execution of the first step leads to reduction of the number of regions and subsequently evaluation time and storage requirement. Execution of the second step leads to 1) construction of a tree which naturally reduces complexity and 2) the constructed tree which deals with a less number of regions.

6. Example

Consider the linear system [13] with $A = \begin{bmatrix} 0.755 & 0.680 \\ 0.651 & -0.902 \end{bmatrix}$, $B = \begin{bmatrix} 0.825 \\ -0.139 \end{bmatrix}$ subject to constraints $|x_i| \leq 10$ for $i=1, 2$ and $|u| \leq 1$. The MPC problem (3) was formulated with $N = 10, Q = 1$ and $R = 1$. The explicit solution (6) is obtained such that (4) is first reformulated into (5) as a QP and then it was solved according to *Theorem 1*

Table 1. Performance of the proposed approach for mp-QP solution.

	Proposed approach	Algorithm from [13]	Algorithm from [14]
Storage (real numbers)	204	425	1266
Arithmetic operations	360	574	4625

The main idea is to combine the binary search tree and the approach removing saturated regions, that a tree was constructed only over the unsaturated regions. The construction resulted in a tree which has a depth 4, and consists of 72 nodes. Sixty eight unique hyper planes occur in the tree. It is clear that implementation of the tree has less complexity in terms of memory reduction and computation time. Table 1 presents the comparison between the proposed approach and the algorithms proposed by [8] in terms of the required storage and FLOPS in order to compute the control input. The comparison shows merit of the proposed approach in terms of memory requirement and computation time. On the other hand, the generality of the approach is more than the methods presented in [13] and [14] that provides an efficient implementation of MPC.

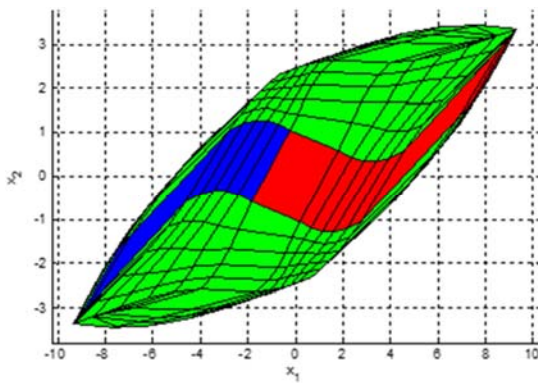


Figure 2. State space partition: polytopic sets $\bar{R}_{sat}, -R_{sat}$ (green), R_{unsat} (blue and red).

7. Conclusions

This paper addressed the important aspects of implementation of explicit MPC in terms of memory requirement and computation time. The approaches search binary tree and removing saturated regions via separator

using MPT Toolbox 3.0 [20]. The solution of the form (6) resulted in 225 regions i.e., $n_r = 225$ which consists of 29 unsaturated regions R_{unsat} , 98 regions \bar{R}_{sat} where $u(x) = 1$, and 98 regions \underline{R}_{sat} with $u(x) = -1$ that the separator function $\xi(x) := -x_1 - x_2 - 0.0011x_1^3 - 0.254x_2^3$ makes the saturated regions \bar{R}_{sat} and \underline{R}_{sat} distinct. The saturated regions are removed using and moreover many regions are removed using the symmetric property, being demonstrated graphically in Figure 2.

On the other hand, a binary search tree was constructed over $n_r = 225$ regions that included in 446 unique hyper planes, 422 candidates for separating hyper planes and depth of 12, consisting 925 nodes. The offline computations to generate the tree was done in 191 seconds, using Matlab 8.3 on a 2.4 GHz Core i5.

function were investigated as the two important approach in implementation of explicit MPC, and their impact on complexity reduction were highlighted. We have described a unified approach which without difficulty combines the approaches search tree and complexity reduction via removing saturated regions such that first, the saturated regions have been removed, and then a tree over the unsaturated regions has been constructed. Merits of the proposed approach and its impact on complexity reduction were demonstrated in a numerical study. The comparative study shows merit of the proposed approach in terms of memory requirement and computation time. On the other hand, the generality of the approach is more than the methods presented in [13] and [14] that provides an efficient implementation of MPC.

References

- [1] Oberdieck, R., Dangelakis, N. A. and Pistikopoulos, E. N. (2017). Explicit model predictive control: a connected-graph approach. *Automatica*, vol. 76, p. 103-12.
- [2] Cimini, G. and Bemporad, A. (2017). Exact Complexity Certification of Active-Set Methods for Quadratic Programming. *IEEE Transactions on Automatic Control*, Apr 24.
- [3] Kvasnica, M. (2016). Implicit vs. explicit MPC—Similarities, differences, and a path towards a unified method. In *Control Conference (ECC) IEEE, 2016 European*, p. 603-603.
- [4] Wen, C., Ma, X. and Ydstie, B. E., (2009). Analytical expression of explicit MPC solution via lattice piecewise-affine function. *Automatica*, vol. 45, no. 4, p. 910-917.
- [5] Di Cairano, S., Yanakiev, D., Bemporad, A., Kolmanovsky, I. V. and Hrovat, D. (2008). December. An MPC design flow for automotive control and applications to idle speed regulation. In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, p. 5686-5691.

- [6] Beccuti, A. G., Papafotiou, G., Frasca, R. and Morari, M. (2007). Explicit hybrid model predictive control of the dc-dcBoost converter. In Power Electronics Specialists Conference, 2007. PESC 2007. IEEE, p. 2503-2509.
- [7] Alessio, A. and Bemporad, A. (2009). A survey on explicit model predictive control. In Nonlinear model predictive control, Springer Berlin Heidelberg, p. 345-369.
- [8] Jones, C. N., Barić, M. and Morari, M. (2007). Multiparametric linear programming with applications to control. European Journal of Control, vol. 13, no. 2-3, p. 152-170.
- [9] Johansen, T. A. and Grancharova, A. (2003). Approximate explicit constrained linear model predictive control via orthogonal search tree. IEEE Transactions on Automatic Control, vol. 48, no. 5, p. 810-815.
- [10] Bemporad, A., Oliveri, A., Poggi, T. and Storace, M. (2011). Ultra-fast stabilizing model predictive control via canonical piecewise affine approximations. IEEE Transactions on Automatic Control, vol. 56, no. 12, p. 2883-2897.
- [11] Storace, M., Repetto, L. and Parodi, M. (2003). A method for the approximate synthesis of cellular non-linear networks—Part 1: Circuit definition. International Journal of Circuit Theory and Applications, vol. 31, no. 3, p. 277-297.
- [12] Kvasnica, M. and Fikar, M. (2012). Clipping-based complexity reduction in explicit MPC. IEEE Transactions on Automatic Control, vol. 57, no. 7, p. 1878-1883.
- [13] Kvasnica, M., Hledík, J., Rauová, I. and Fikar, M. (2013). Complexity reduction of explicit model predictive control via separation. Automatica, vol. 49, no. 6, p. 1776-1781.
- [14] Tøndel, P., Johansen, T. A. and Bemporad, A. (2003). Evaluation of piecewise affine control via binary search tree. Automatica, vol. 39, no. 5, p. 945-950.
- [15] Gupta, A., Bhartiya, S. and Nataraj, P. S. V. (2011). A novel approach to multiparametric quadratic programming. Automatica, vol. 47, no. 9, p. 2112-2117.
- [16] Bemporad, A., Morari, M., Dua, V. and Pistikopoulos, E. N. (2002). The explicit linear quadratic regulator for constrained systems. Automatica, vol. 38, no. 1, p. 3-20.
- [17] Johansen, T. A., Jackson, W., Schreiber, R. and Tondel, P. (2007). Hardware synthesis of explicit model predictive controllers. IEEE Transactions on Control Systems Technology, vol. 15, no. 1, p. 191-197.
- [18] Mariéthoz, S., Mäder, U. and Morari, M. (2009). High-speed FPGA implementation of observers and explicit model predictive controllers. In IEEE IECON, Ind. Electronics Conf., Porto, Portugal, Nov 2009.
- [19] Bayat, F., Johansen, T. A. and Jalali, A. A. (2012). Flexible piecewise function evaluation methods based on truncated binary search trees and lattice representation in explicit MPC. IEEE Transactions on Control Systems Technology, vol. 20, no. 3, p. 632-640.
- [20] Herceg, M., Kvasnica, M., Jones, C. N. and Morari, M. (2013). Multi-parametric toolbox 3.0. In Control Conference (ECC), 2013 European, p. 502-510.