Real-time suboptimal Model Predictive Control using a combination of Explicit MPC and Online Optimization

Melanie N. Zeilinger, Colin N. Jones and Manfred Morari

Automatic Control Laboratory, ETH Zurich, Physikstrasse 3, ETL I 28, CH – 8092 Zurich, Switzerland
zeilinger | cjones | morari@control.ee.ethz.ch

***Abstract*— Limits on the storage space or the computation time restrict the applicability of model predictive controllers (MPC) in many real problems. Currently available methods either compute the optimal controller online or derive an explicit control law. In this paper we introduce a new approach combining the two main paradigms of explicit and online MPC to overcome their individual limitations. The algorithm computes a piecewise affine approximation of the optimal solution that is used to warm-start an active set linear programming procedure. A preprocessing method is introduced that provides hard real-time, stability and performance guarantees for the proposed controller. By choosing a combination of the quality of the approximation and the number of online active set iterations the presented procedure offers a tradeoff between the warm-start and online computational effort. We show how the problem of identifying the optimal tradeoff for a given set of requirements on online computation time, storage and performance can be solved. Finally, we demonstrate the potential of the proposed warm-start procedure on a numerical example.**

I. Introduction

It is well-known that computation of a model predictive controller (MPC), under certain assumptions on the problem structure, amounts to the solution of a linear or a quadratic program at each sampling instant. Whereas classically, these optimization problems are solved online, it has been shown in recent years that the optimal solution to this type of problem is a piecewise affine function (PWA) defined over a polyhedral partition of the feasible states. This so-called explicit solution can then be used as a control look-up table online, enabling MPC to be used for fast sampled systems.

However, both the offline and online method have limitations. The main disadvantage of the online method is that it is in general only applicable for controlling slow processes. In the case of the explicit solution, the number of state-space regions over which the control law is defined, the so-called complexity of the partition, grows in the worst case exponentially due to the combinatorial nature of the problem. This has given rise to an increasing interest in the development of new methods to either improve online optimization or to approximate explicit solutions (e.g. [4], [21], [22]). Depending on the particular problem properties and implementation restrictions, the user then has to decide for one of the two approaches.

This work aims at enlarging the possibilities to tradeoff solution properties through the combination of these two methods. Specifically, ideas from approximation are combined with warm-start techniques. In this paper we use a PWA approximation of the optimal control law, which has been computed offline, to warm-start the online optimization. The optimization executes a finite number of active set iterations before returning a feasible, suboptimal control action, which is then applied to the system. The idea is to choose a good tradeoff between the complexity of the PWA approximation and the number of active set iterations required in order to satisfy system constraints in terms of online computation, storage and performance. Conditions are derived which guarantee that the suboptimal solution is closed-loop stabilizing, feasible and has a bounded performance deterioration.

We also raise the question of an optimal tradeoff, meaning that the best combination of warm-start and online computational effort is chosen, with respect to certain requirements on the solution. Considering computation time and performance as two exemplifying requirements, this can be informally stated in the form of the following optimization problems:

1. Minimize online computation time while respecting a bound on the performance deterioration
2. Maximize the performance within available computation time

The outline of the paper is as follows: In Section IV we introduce the main idea of using an offline approximation to warm-start an active set linear programming procedure. An explicit representation of the proposed control law is derived in Section V. In Section VI a preprocessing method is introduced that allows the analysis of the properties of the control input that will be applied online. The question of an optimal tradeoff between warm-start and online computation is discussed in Section VII. Finally, a numerical example illustrating the proposed method and ideas is given in Section VIII.

II. Notation

If $A \in \mathbb{R}^{m\times n}$ and $I \subseteq \{1, \cdots, m\}$, then $A_I \in \mathbb{R}^{|I|\times n}$ is the matrix formed by the rows of A indexed by I. If $c \in \mathbb{R}^m$ is a vector then c_I is the vector formed by the elements of c indexed by I. Any set $B \subseteq \{1, \cdots, m\}$ such that $|B| = n$ and $\text{rank}(A_B) = n$ is called a *basis*.

A *polyhedron* is the intersection of a finite number of

halfspaces $P = \{x|Ax \leq b\}$ and a *polytope* is a bounded polyhedron. $\mathcal{P}^N := \{P_j\}_{j=1}^N$ with $N \in \mathbb{N}$ is called a polyhedral partition of $\mathcal{X} \subseteq \mathbb{R}^d$ if all P_j are full-dimensional polyhedra, $\cup_{j=1}^N P_j = \mathcal{X}$ and $\text{int}P_i \cap \text{int}P_j = \emptyset$, $\forall i \neq j$ and $i, j \in \{1, \cdots, N\}$. A PWA function $f(x)$ defined over the polyhedral partition $\mathcal{P}^N$ is denoted as $f(x) = \{C^j x + D^j | x \in P_j, P_j \in \mathcal{P}^N\}$.

III. Preliminaries

Model Predictive Control

In this paper, we consider the following formulation of the MPC problem:

Problem III.1 (MPC problem).

$$J^*(x) = \min_{\{u_0,\ldots,u_{N-1}\}} V_N(x_N) + \sum_{i=0}^{N-1} l(x_i, u_i) \tag{1}$$
$$\text{subject to} \quad \begin{array}{ll} x_{i+1} & = A_d x_i + B_d u_i \ , \\ (x_i, u_i) & \in \mathbb{X} \times \mathbb{U} \ , \\ x_N & \in \mathcal{X}_F \ , \\ x_0 = x \ , & \end{array}$$

where $\mathbb{X}, \mathbb{U}$ and $\mathcal{X}_\mathcal{F}$ are polytopic constraints on the states and inputs, the stage cost is defined as $l(x_i, u_i) := \|Qx_i\|_p + \|Ru_i\|_p$ with $p \in \{1, \infty\}$, V_N is the terminal penalty function and $\mathcal{X}_F \subseteq \mathbb{X}$ is a compact terminal target set, with properties as defined in Assumption III.3. The MPC problem implicitly defines the set of feasible initial states $\mathcal{X}$.

Definition III.2 (Feasibility). A control law $u(x)$ is called feasible for x if $(x, u(x)) \in \mathbb{X} \times \mathbb{U}$.

Assumption III.3 (Stability). In the following it is assumed that the parameters N, Q, R, V_N, and $\mathcal{X}_F$ are chosen in such a way that problem (1) generates a feasible and stabilizing control law for all $x \in \mathcal{X}$ when applied in a receding horizon fashion and $J^*(x)$ is a polyhedral PWA Lyapunov function.

If the norm p is taken to be the $1-$ or the $\infty-$norm, we can write (1) as a parametric Linear Program (pLP) of the form:

$$u^*(x) = \ \arg\min_u \ c^T u \tag{2}$$
$$\text{subject to} \quad \begin{array}{l} A_I u \leq b_I \ , \\ A_E u = B_E x \ , \end{array}$$

where $u \in \mathbb{R}^n$ is a vector containing the sequence of control inputs $\{u_0, \ldots, u_{N-1}\}$ and appropriate slack variables, the current state $x \in \mathbb{R}^d$ is the parameter, $A \in \mathbb{R}^{m \times n}$, $E \subset \{1, \cdots, m\}$ and $I = \{1, \cdots, m\} \setminus E$ (see e.g. [8] for details on the conversion). (Note that for simplicity we use the same indexing for b and B as for A although we distinguish the vector b from the matrix B in order to account for the different dimension). By solving the pLP (1) the optimal control input $u^*(x)$ can be computed for each feasible value of the state.

Theorem III.4 (Solution to the MPC problem, [2]). *The MPC control law is a PWA function of the state x defined over a polyhedral partition of the feasible set $\mathcal{X}$.*

A. Approximation of the MPC problem

We first define an approximation of the MPC problem (1) and some useful properties that will be used in Section VI in order to give guarantees on the control law proposed in this paper. Let $u^*(x)$ be the optimizer of the optimal control problem (1) and $J^*(x) = c^T u^*(x)$ be the corresponding optimal cost and a Lyapunov function.

Definition III.5. A function $\widetilde{u}(x)$ is called an *approximate control law* for (1) if

1. $\widetilde{u}(x)$ is feasible for all $x \in \mathcal{X}$
2. $\widetilde{J}(x) := c^T \widetilde{u}(x) \geq J^*(x)$ for all $x \in \mathcal{X}$

The *approximation error* is defined by

Definition III.6 (Approximation error). $\widetilde{u}(x)$ is an ϵ approximation if for all $x \in \mathcal{X}$: $\widetilde{J}(x) - J^*(x) \leq \epsilon$.

A standard condition to test if an approximate solution is *stabilizing* is given by the following theorem:

Theorem III.7 (Stability, [18]). *If $\widetilde{u}(x)$ is an approximate control law of* (1)*, then $\widetilde{J}(x)$ is a Lyapunov function for the system $x_{i+1} = A_d x_i + B_d \widetilde{u}(x_i)$ if the approximation error ϵ is less than $\min_{u \in \mathbb{U}}\{l(x, u)\}$ for all $x \in \mathcal{X}$, where $l : \mathbb{R}^d \times \mathbb{R}^m \to \mathbb{R}$ is the stage cost (1).*

IV. Proposed control law

In order to overcome the limitations of the offline and online methods mentioned in the introduction, several authors recently proposed new approaches to speed up online optimization or to reduce the complexity of explicit solutions by means of approximation. The authors in [16] for example utilize new developments in interior-point methods and show how these can be applied to efficiently solve the optimal control problem. Another paradigm that is frequently applied to improve online optimization is warm-starting (see e.g. [5], [22]). In explicit MPC, approximation methods have been proposed that either modify the original MPC problem (1), retrieve a suboptimal solution or postprocess the computed optimal solution, with the goal of reducing the complexity of the explicit controller, cf. e.g. [3], [4], [14]. Most proposals concerning online as well as offline approaches however lack the possibility of giving guarantees on the suboptimal solution, e.g. closed-loop stability.

The strategy proposed in this paper combines the idea of offline approximation with warm-start techniques from online optimization with the goal of providing hard real-time, stability and performance guarantees. Warm-start techniques aim at identifying advanced starting points for the optimization in order to reduce the number of iterations required to reach the optimum. They often try to make use of the information gained during the solution of one problem to solve the next one in a sequence of closely related problems. When solving MPC problems in a receding horizon fashion an LP is computed for every measured state. However, the optimal control input from a previous measurement might

be an infeasible solution to (1) at the current instance. We therefore propose a warm-start strategy that utilizes a PWA approximate control law of (1) to provide a good and feasible starting point. The pre-knowledge of the initial solution for all feasible values of x allows us to analyze the solution obtained by the online optimization. The following two parameters are used to classify the warm-start solution: the complexity N_P (number of regions) and its approximation error ϵ, given by Definition III.6.

Definition IV.1 (Warm-start Solution). A function $\nu(x, N_P)$ is called a warm-start solution of (1) if it is a feasible, PWA approximate control law of (1) defined over N_P polytopic regions.

Lemma IV.2 (Convergence of $\nu(\cdot,\cdot)$). *There exists a function $\nu(x, N_{P_{opt}})$ of finite complexity $N_{P_{opt}} \in \mathbb{N}$, such that $u^*(x) = \nu(x, N_{P_{opt}})$ for all $x \in \mathcal{X}$, where $u^*(x)$ is the optimal solution to* (2).

Proof. Result follows directly from the fact that the approximation is PWA and Theorem III.4. ■

By means of the parameter N_P requirements can be set on the complexity of the warm-start solution $\nu(\cdot, N_P)$ that is computed and stored in an offline preprocessing step.

Remark IV.3. For available approximation methods there exists a relation between the approximation error ϵ and the complexity N_P of an approximation. Requirements on the approximation error can therefore also be imposed using the parameter N_P. This will be discussed in detail in Section VII.

In the online control procedure, the warm-start solution is evaluated for the measured state and used to initialize the online optimization. A standard active set method (ASM) is applied to compute the control action since it allows us to take advantage of a feasible starting point. Active set methods generate a sequence of feasible iterates that converge to the optimal solution. At each iterate u, the active set is given by $I_A = E \cup \{i \in I | A_i u = b_i\}$. In an active set iteration, a subset of the active set is chosen as the working set W using standard heuristics. From the current iterate u, the maximal step in the steepest descent direction is then computed, which is the direction that minimizes the objective in (2) while keeping the constraints in W active, defining the next iterate. See e.g. [10], [12] for more details on active set methods. Whereas in standard active set methods iterations are performed until the optimality conditions are met, the online optimization procedure is stopped early after exactly K active set iterations and the current suboptimal control input is applied to the system.

Definition IV.4 (Warm-start optimization). Let u be a feasible solution of (2) for the parameter x. We define $\kappa(u, K)$ to be the decision variable of (2) after K iterations of the linear programming active set method.

Definition IV.5 (Proposed control law). Let $\nu(\cdot, N_P)$ be an approximate PWA solution to (2) and $\kappa(\cdot, K)$ be as defined in IV.4. The proposed control law is

$$u_{\text{on}}(x) = \kappa(\nu(x, N_P), K), \text{ for } x \in \mathcal{X} \ . \tag{3}$$

Lemma IV.6 (Properties of $\kappa(\cdot,\cdot)$). *The proposed control law* (3) *is feasible for all $x \in \mathcal{X}$, and for each N_P there exists a finite $K_{opt} \in \mathbb{N}$, such that $u^*(x) = \kappa(\nu(x, N_P), K_{opt})$.*

Proof. Feasibility is ensured by the procedure of the ASM and the fact that $\nu(x, N_P)$ is feasible for all $x \in \mathcal{X}$. The existence of a finite K_{opt} is guaranteed by the convergence of the ASM in finite time [10]. ■

The warm-start linear programming procedure for a fixed complexity N_P and number of iterations K is summarized in Algorithm 1.

Algorithm 1 Warm-start linear programming procedure

Input: warm-start solution $\nu(\cdot, N_P)$ and current measured state x_{meas}
Output: approximate control input u_{on}

```
run point location algorithm: u = ν(x_meas, N_P)        [6]
for k = 1, ··· , K do
    perform an active set iteration; update iterate u   [12]
end for
u_on = u
```

In Section VI an offline analysis is introduced providing guarantees for the proposed control law $u_{\text{on}}(x)$ in (3) to be closed-loop stabilizing, feasible and to have a bounded performance deterioration compared to the optimal solution.

The above described procedure of using an approximation to warm-start an online optimization offers the possibility to decide on the complexity and approximation error of the warm-start solution $\nu(\cdot,\cdot)$. A tradeoff can be made between the degree of approximation realized by the warm-start and the effort expended in online optimization. The goal is to identify a good if not optimal tradeoff that achieves the best properties of the online control input applied to the system for given requirements on the approximation error and/or limitations on the online computation time or storage.

The proposed procedure and algorithms are detailed in the following sections.

V. Parametric calculation of the online control law

Our goal is to give guarantees on the proposed suboptimal control law (3). Apart from feasibility, which is guaranteed by Lemma IV.6, we want to ensure stability and a certain bound on the approximation error. In order to analyze the solution properties, we need an explicit representation of the approximate control input $\kappa(\cdot,\cdot)$ for the entire feasible set $\mathcal{X}$. We will show that starting from the warm-start solution, the iterative path taken by the active set method is a function of x, defined over a polyhedral subdivision of $\mathcal{X}$.

Remark V.1 (Offline Complexity). Note that the complexity of this subdivision does not affect the actual optimization carried out online, since the parametric solution is only used for offline analysis.

The operations performed during an active set iteration can be formulated as functions of the parameter x. Let $u^k(x)$ be the iterate and W the working set at the k-th iteration step. Applying the KKT conditions we get that $\triangle u$ is the steepest descent direction from $u^k(x)$ if and only if there exists a vector $\triangle\lambda$ such that $(\triangle u, \triangle\lambda)$ satisfies the following system:

$$\begin{bmatrix} I & A_W^T \\ A_W & 0 \end{bmatrix} \begin{bmatrix} \triangle u \\ \triangle\lambda \end{bmatrix} = \begin{bmatrix} -c \\ 0 \end{bmatrix} \tag{4}$$

The line search determining the step length to the best feasible point along $\triangle u$ is given by

$$\tau(x) = \min_{i \in I \setminus I_W} \{ \frac{b_i - A_i u^k(x)}{A_i \triangle u} | A_i \triangle u > 0 \} \ . \tag{5}$$

The new active constraint is $A_e u^k(x) = b_e$, where e is the optimizer of (5). The next iterate is determined by

$$u^{k+1}(x) = u^k(x) + \tau(x) \triangle u \ . \tag{6}$$

Theorem V.2. *At every iteration $k \in \{1, \cdots, K\}$, the step length $\tau(x)$ in* (5) *and the current iterate $u^k(x)$ are PWA functions of x defined over a polyhedral partition $\mathcal{G}^{N_k}$ of the feasible set $\mathcal{X}$.*

Proof. Assume that the statement is true for k and that $u^k(x) := \{C^j x + D^j | x \in G_j, G_j \in \mathcal{G}^{N_k}\}$. For one region $\mathcal{G}_j \in \mathcal{G}^{N_k}$ the line search (5) determining the next constraint to become active is given by the pLP $\tau(x) = \min_i \{\alpha_i(x)\}$, with $\alpha_i(x) = \frac{-A_i C^j}{A_i \triangle u} x + \frac{b_i - A_i D^j}{A_i \triangle u}$, $i \in I_{cand}$ and $I_{cand} = \{i \in I \setminus I_A | A_i \triangle u > 0\}$. This shows that $\tau(x)$ is a PWA function of x, since the optimal cost of a pLP is PWA [2], [11]. With $\tau(x)$ and $u^k(x)$ being PWA, the $k+1$-th iterate (6) is as well a PWA function of x. Since the initial solution is $u^0(x) = \nu(x, N_P)$, the statement is true for $k = 1$ and hence for all $k \in \{1, \cdots, K\}$. ■

With $u^k(x)$ being a PWA function, equation (5) results in a parametric LP. The approximate control law at iteration K is obtained by solving (4) and (5) iteratively for all $k \in \{1, \cdots, K\}$. With each iteration, the parametric solution of (5) causes a further refinement of the polyhedral partition $\mathcal{G}^{N_k}$.

Remark V.3. Note that the computation of the PWA function $\tau(x)$ can be reduced to redundancy elimination.

Corollary V.4. *The proposed control law $u_{on}(x)$ is a PWA function of x defined over a polyhedral partition $\mathcal{G}^{N_K}$ of the feasible set $\mathcal{X}$.*

Theorem V.2 enables us to compute an explicit PWA representation of the approximate control law $\kappa(\nu(x, N_P), K)$ for a fixed complexity N_P and number of iterations K using Algorithm 2.

Algorithm 2 Offline Analysis

Input: warm-start solution
$\nu(x, N_P) = \{\nu_j(x) \,|\, x \in P_j, P_j \in \mathcal{P}^{N_P}\}$
Output: Explicit representation of the proposed control law $u_{\text{on}}(x)$

```
initialize stack S = ∅
for all P_j ∈ 𝒫^{N_P} do push (ν_j(x), P_j) onto S
end for
for all k ∈ {1, ···, K} do        [K active set iterations]
    initialize stack Ŝ = ∅
    while S ≠ ∅ do                [subdivide each region]
        pop (u_P(x), P) from S
        compute △u and τ(x) for u_P(x)        (4),(5)
        let τ(x) = {f_j(x) | x ∈ R_j, R_j ∈ ℛ}
        for all R_j ∈ ℛ do
            push (u_P(x) + f_j(x)△u, R_j) onto Ŝ
        end for
    end while
    u^k(x) := {u_j(x) | x ∈ P_j, (u_j(x), P_j) ∈ Ŝ}
    S = Ŝ
end for
u_on(x) = u^K(x)
```

VI. Analysis

We introduce a preprocessing analysis that investigates the following properties of the approximate control law $\kappa(\nu(x, N_P), K)$: stability, approximation error, storage space and online computation time.

Lemma VI.1 (Approximation error of $u_{\text{on}}(x)$). *If $u_{on}(x) := \{C^j x + D^j | x \in P_j, P_j \in \mathcal{P}^{N_K}\}$ is the proposed control law in* (3)*, then the approximation error defined in Definition III.6 is given by*

$$\epsilon_K = \max_j \{d_j\} \ , \ \textit{with} \tag{7}$$

$$d_j = \max_{u,x} \{c^T (C^j x + D^j) - c^T u \,|\, u \in \mathbb{U}, x \in P_j\} \ , \tag{8}$$

for all $j \in \{1, \cdots, N_K\}$.

Proof. The biggest error over all $x \in \mathcal{X}$ and hence over all the regions in $\mathcal{P}^{N_K}$ is the smallest ϵ that fulfills the condition in Definition III.6 for all $x \in \mathcal{X}$. ■

Stability can be easily tested using the conditions of Theorem III.7.

Storage space is determined by the complexity (number of regions) N_P of the warm-start solution since only the warm-start has to be stored.

Online computation time will be estimated in terms of floating point operations (flops) for the calculations that have to performed online. First the region of the current state is identified using the point location algorithm in [7], then the corresponding affine control law is evaluated and finally K online iterations are executed.

Remark VI.2 (Sparsity of the MPC problem). In the case of the MPC problem (1), the matrices A and B in

(2) have a special structure, resulting from the particular problem setup. The matrices are extremely sparse and by reordering can be shown to be in fact block diagonal or banded. We can exploit the banded structure of the matrices to solve equations (4) and (5), achieving significant computational savings [21].

Theorem VI.3 (Flop count). *If the input and state dimensions are m and d respectively and the number of constraints on each state-input pair is m_c, then the number of flops to calculate the control action $u_{on}(x_{meas})$ for a measured state x_{meas} can be bounded by:*

$$f_{on} = N_P f_{ws} + K f_{ASM} \ , \tag{9}$$
$$\text{where} \quad f_{ws} = 2d \ ,$$
$$f_{ASM} = N((11d+8m)^2 + 4m_c(11d+8m) - d^2 + md - 30(m+d) - 3m_c)$$

f_{ws} denotes the flop number for point location per region, f_{ASM} the flops per active set iteration and N the horizon in (1).

Proof. The number of flops for point location is given in [7]. The flop counts for active set iterations use the fact that the matrices in (2) have banded structure. An LU factorization and LU updates as described in [21] is considered, where L and U have half-bandwidth $3d+2m+m_c-1$. The worst case is taken in form of the maximal number of active constraints before a full basis is encountered. The same bound can be used for the case of a full basis since the iterations then equal simplex steps that, using sparse rank one LU updates (see e.g. [19]), result in a lower flop number. The flop counts for calculation of the steepest descent direction, step length and the next iterate follow directly from the equations (4), (5) and (6). ■

Remark VI.4. The flop counts in Theorem VI.3 are based on a 1-norm in (1). Flop counts for the ∞-norm can be derived similarly. Since the ∞-norm results in a smaller number of flops due to the lower number of variables we can however also use the bound derived in Theorem VI.3.

The worst-case estimates for the properties of the proposed control law $\kappa(\cdot, K)$ in terms of stability, approximation error, storage space and online computation time can be calculated. For fixed parameters N_P and K this allows us to give guarantees on the properties of the control law that is applied to the system.

Remark VI.5. Note that the computational effort for the analysis can be reduced by applying ideas from search trees. A node in the tree represents a single region and each depth level corresponds to an active set iteration. Using depth-first-search, for instance, an error bound for the K-th active set iteration can be derived without calculating the full parametric solution.

VII. Optimization over the parameters

In this section we will now try to optimize over the parameters that determine the applied control input: the complexity N_P of the warm-start solution and the number of iterations K. The choice of the warm-start solution $\nu(\cdot, N_P)$ determines the computational effort for point location f_{ws} in (9) on the one hand and the quality of the warm-start on the other and therefore the number of active set iterations. This offers the possibility to trade off the amount of online computation time spent on the warm-start with that spent on online optimization. The challenge is to identify an optimal tradeoff that achieves the best properties of the applied control input.

We consider the problem of optimizing the online time to compute a control law that guarantees stability, a certain performance bound ϵ_{max} and a limit on the storage space $N_{P,max}$. Computation time is again measured in the form of flops (9), resulting in the following optimization problem

$$F_{min} = \min_{N_P, K} N_P f_{\text{ws}} + K f_{\text{ASM}} \tag{10}$$
$$\text{subject to} \quad \epsilon_K \leq \epsilon_{max} \ , \quad N_P \leq N_{P,max} \ .$$

In order to solve this optimization problem, a method to generate an approximation of (1) with complexity N_P needs to be defined. There are several approximation methods that can be used to create a PWA warm-start solution (e.g. [4], [15], [20]).

In this work the method introduced in [14] was chosen, which is based on the beneath/beyond (B/B) algorithm, a common approach for convex hull calculation [1], [13]. An approximation $\widetilde{J}(x)$ of $J^*(x)$ in (2) is constructed by computing the convex hull of a subset of vertices of the epigraph of $J^*(x)$. The approximation can be iteratively improved by adding one vertex at a time and updating the convex hull. When all vertices of the polytope are included, the optimal solution of (2) is reached. The approximate control law is obtained by interpolating between the optimal control inputs at the vertices. The main advantage of the beneath/beyond method is that it is an incremental approach, allowing one to set requirements on either the complexity or the error of the approximation $\widetilde{J}(x)$. In addition, it is based on an implicit rather than on an explicit representation of the optimal solution and is hence not dependent on the computability of the optimal parametric solution to pLP (2).

Theorem VII.1 (B/B warm-start solution [14]). *Given a parameter $N_P \in \mathbb{N}$ the B/B method returns a feasible PWA approximation $\nu(\cdot, N_P)$ of* (2).

Remark VII.2. The approximate control law generated by the B/B method is not necessarily defined over the entire feasible set. For the examples given in this paper, the B/B algorithm was initialized with all the vertices of the boundary of $\mathcal{X}$ in order to provide a feasible control law for all $x \in \mathcal{X}$. However, it is possible to extend the method so that it works for a subset of $\mathcal{X}$.

The error of a B/B approximation is related to its complexity by the following Lemma VII.3.

Lemma VII.3 (Complexity/Error, [9], [17]). *Let $\nu(\cdot, N_P)$ be an approximation to* (2) *generated by the B/B method, ϵ_{BB} its approximation error and N_P its complexity. For every ϵ_{BB} there exists an $N_{BB} \in \mathbb{N}$ such that the approximation error of $\nu(\cdot, N_{BB})$ is less than ϵ_{BB}.*

Remark VII.4. Note that whereas the approximation error of a B/B approximation is monotonically decreasing with every B/B improvement, the complexity might not be monotonic (see [14]).

Using a B/B approach, problem (10) is a function of only the complexity N_P of the warm-start solution. This follows from the fact that for each complexity N_P of the B/B warm-start there exists exactly one number of iterations K to achieve a certain approximation error ϵ_{max}. The optimal tradeoff problem (10) can therefore be solved using the B/B approach. Since the calculation of B/B approximations can be computationally expensive, we propose to solve a subproblem instead. We use a selection of values for N_P and solve the restricted minimization problem (10) for each of them. The best tradeoff is represented by the solution with the minimum cost value. The samples can then be iteratively refined to improve the obtained result.

Remark VII.5. The problem of identifying the optimal tradeoff to minimize the approximation error subject to constraints on the online computation time and the storage space can be solved following the same procedure described for problem (10).

VIII. Numerical Example

In this section we will demonstrate the advantages of the proposed warm-start linear programming procedure. We exemplify the main procedure of solving the optimal tradeoff problem for an example. Consider the randomly generated system:

$$x_{i+1} = \begin{bmatrix} -0.251 & 0.114 & 0.123 & -0.433 \\ 0.319 & -0.658 & 0.905 & 0.118 \\ 0.459 & -0.484 & -0.175 & -0.709 \\ 0.016 & 0.116 & -0.002 & -0.505 \end{bmatrix} x_i + \begin{bmatrix} -0.873 & 0.879 \\ 0.669 & 0.936 \\ -0.353 & 0.777 \\ 0.268 & -0.336 \end{bmatrix} u_i \ , \tag{11}$$

with a prediction horizon $N = 5$ and the constraints $\|u\|_\infty \leq 5$, $\|x\|_\infty \leq 5$ on the input and state respectively. The norm p for the stage cost is taken as the ∞-norm and the weight matrices Q and R are taken as the identity matrix and two times the identity matrix.

The restricted optimal tradeoff problem (10) was solved for a set of warm-start solutions with $N_P \in \{7003, 10001, 12009, 15001\}$ and a maximum approximation error $\epsilon_{max} = 2$ that corresponds to a performance deterioration of about 0.1%, taken over a large number of sample points. The performance deterioration is measured as the relative difference between the cost of the closed loop trajectory using the optimal control input and the one using the suboptimal control input, given by $\left[\sum_{i=0}^{\infty} \left(l(x_i, u_{\text{on}}(x_i)) - l(x_i, u^*(x_i))\right)\right] / \sum_{i=0}^{\infty} l(x_i, u^*(x_i))$. The results for the different values of N_P are shown in Figure 1. The best tradeoff between approximation and online iterations is given by the lower envelope of the curves, as it represents the best online computation time for a certain approximation error (or the other way round). For an approximation error up to 3.4, a pure approximation by the B/B method results in the fastest computation times. For any error below 3.4 a combination of a warm-start solution of complexity $N_P = 15001$ with active set iterations represents the best tradeoff. Note that a further improvement of the warm-start solution does not improve the results. In the case of additional storage limitations, one can also choose a warm-start solution of lower complexity resulting in only slightly higher computation times. The optimal strategy is therefore often not to achieve the best warm-start solution, but a particular combination of warm-start and online optimization. In comparison with a pure online solution, the warm-start procedure is always superior. The solution to the optimal tradeoff problem does however highly depend on the particular problem structure. For certain problems the best solution method will be a particular combination of the two methods whereas for others it will as well be a pure offline or online approximation, e.g. in the case of extremely small or large problems.

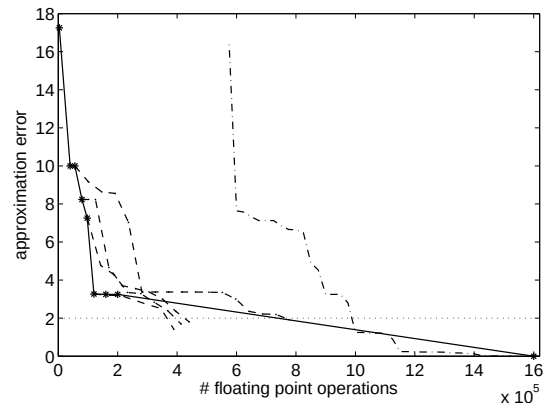

Fig. 1. Warm-start procedure starting from four different PWA B/B approximations with $N_P \in \{7003, 10001, 12009, 15001\}$. The solid line represents the offline B/B approximations. The flop number for zero approximation error was interpolated, since the optimal solution could not be computed. The four dashed lines show the improvement by the active set method warm-started from each of the four approximations. The dash-dotted line is a sampled worst-case estimate of a pure online solution using the Simplex method, shown after the first feasible solution is found.

References

[1] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software*, vol. 22, no. 4, pp. 469–483, 1996.

[2] A. Bemporad, F. Borrelli, and M. Morari, "Model predictive control based on linear programming - the explicit solution," *IEEE Transactions on Automatic Control*, vol. 47, no. 12, pp. 1974–1985, 2002.

[3] A. Bemporad and C. Filippi, "Suboptimal explicit MPC via approximate multiparametric quadratic programming," in *Proceedings of the 40th IEEE Conference on Decision and Control*, vol. 5, 2001, pp. 4851–4856.

[4] ——, "An algorithm for approximate multiparametric convex programming," *Comput. Optim. Appl.*, vol. 35, no. 1, pp. 87–108, 2006.

[5] H. G. Bock, M. Diehl, D. B. Leinweber, J. P. Schlöder, and E. Stein, "Efficient direct multiple shooting in nonlinear model predictive control," in *Scientific Computing in Chemical Engineering II*, F. Keil,

W. Mackens, H. Voss, and J. Werther, Eds. Springer, 1999, vol. 2, ch. 4, pp. 218–227.

[6] F. Borrelli, "Discrete time constrained optimal control," Ph.D. dissertation, Swiss Federal Institute of Technology (ETH), 2002.

[7] F. Borrelli, M. Baotic, A. Bemporad, and M. Morari, "Efficient online computation of constrained optimal control," in *Proceedings of the 40th IEEE Conference on Decision and Control*, vol. 2, 2001, pp. 1187–1192.

[8] F. Borrelli, A. Bemporad, and M. Morari, "A Geometric Algorithm for Multi-Parametric Linear Programming," *Journal of Optimization Theory and Applications*, vol. 118, no. 3, pp. 515–540, Sept. 2003.

[9] E. M. Bronshteyn and L. D. Ivanov, "The approximation of convex sets by polyhedra," *Siberian Mathematical Journal*, vol. 16, no. 5, pp. 852–853, 1975.

[10] R. Fletcher, *Practical Methods of Optimization*, 2nd ed. John Wiley and Sons, New York, 1987.

[11] T. Gal and J. Nedoma, "Multiparametric linear programming," *Management Science*, vol. 18, no. 7, pp. 406–422, 1972.

[12] G. C. Goodwin, M. M. Seron, and J. de Doná, *Constrained Control and Estimation, An Optimisation Approach*. Springer, 2006.

[13] B. Grünbaum, "Measures of symmetry for convex sets," *Proceedings of the Seventh Symposium in Pure Mathematics of the American Mathematical Society, Symposium on Convexity*, pp. 233–270, 1961.

[14] C. N. Jones, M. Baric, and M. Morari, "Multiparametric Linear Programming with Applications to Control," *European Journal of Control*, vol. 13, no. 2-3, pp. 152–170, Mar. 2007.

[15] B. Lincoln and A. Rantzer, "Relaxing dynamic programming," *Automatic Control, IEEE Transactions on*, vol. 51, no. 8, pp. 1249–1260, 2006.

[16] C. V. Rao, S. J. Wright, and J. B. Rawling, "Application of interior–point methods to model predictive control," *Journal of Optimization Theory and Applications*, vol. 99, pp. 723–757, 1998.

[17] R. Schneider and J. A. Wieacker, "Approximation of convex bodies by polytopes," *Bulletin London Mathematical Society*, vol. 13, no. 2, pp. 149–156, March 1981.

[18] P. O. M. Scokaert, D. Q. Mayne, and J. B. Rawlings, "Suboptimal model predictive control (feasibility implies stability)," *IEEE Transactions Automatic Control*, vol. 44, no. 3, pp. 648–654, 1999.

[19] L. M. Suhl and U. H. Suhl, "A fast lu update for linear programming," *Annals of Operations Research*, vol. 43, no. 1, pp. 33–47, 1993.

[20] P. Tondel, T. Johansen, and A. Bemporad, "Computation and approximation of piecewise affine control laws via binary search trees," *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, vol. 3, pp. 3144–3149, 2002.

[21] S. J. Wright, "Applying new optimization algorithms to model predictive control," in *Chemical Process Control-V, CACHE*, ser. AIChE Symposium Series No. 316, vol. 93, 1997, pp. 147–155.

[22] E. A. Yildirim and S. J. Wright, "Warm-start strategies in interior-point methods for linear programming," *SIAM Journal on Optimization*, vol. 12, pp. 782–810, 2002.