Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 53-2 (2020) 6542–6547

Fast Gradient Method for Model Predictive Control with Input Rate and Amplitude Constraints ⋆

Idris Kempf, Paul Goulart & Stephen Duncan *

* *Department of Engineering Science, University of Oxford, UK (e-mail: {idris.kempf, paul.goulart, stephen.duncan}@eng.ox.ac.uk)*

Abstract: This paper is concerned with the computing efficiency of model predictive control (MPC) problems for dynamical systems with both rate and amplitude constraints on the inputs. Instead of augmenting the decision variables of the underlying finite-horizon optimal control problem to accommodate the input rate constraints, we propose to solve this problem using the fast gradient method, where the projection step is solved using Dykstra's algorithm. We show that, relative to the Alternating Direction of Method Multipliers (ADMM), this approach greatly reduces the computation time while halving the memory usage. Our algorithm is implemented in C and its performance demonstrated using several examples.

Keywords: Model Predictive Control (MPC), Fast Gradient Method, Dykstra's Method, Alternating Direction of Multipliers Method (ADMM), Projection, Rate Constraints

1. INTRODUCTION

Despite its advantages for constraint handling and feed-forward disturbance modeling, the applicability of model predictive control (MPC) is limited by the requirement to solve optimization problems in real-time to compute the control law. While optimization problems are easily solved on a standard computer in a simulation environment, it is considerably more challenging to solve such problems on the embedded systems often encountered in industrial applications. Cost factors often outweigh the need for powerful hardware to implement an MPC scheme using standard optimization program solvers. Embedded systems employed in industrial applications, e.g. in the aviation or automotive industry, therefore feature much lower computational power than a standard desktop computer and much tighter memory restrictions. Hardware platforms in these industries commonly run at clock frequencies of a few hundred megahertz, while offering only a few megabytes of memory, whereas standard computers run at gigahertz rates and can provide many gigabytes of memory storage. A key to employing MPC in industrial applications is therefore an algorithm that makes maximum use of available computing resources while keeping the memory usage at a minimum.

Constraints imposed on control systems can be split into state, output and input constraints. State and output constraints are usually associated with safety or operational considerations. Input constraints, however, are enforced by physical limitations of the actuators (Saberi et al., 1999, Chapter 1). It is also observed that two different kinds of constraints are usually imposed on the input: rate and amplitude constraints. This paper focuses on systems for which constraints on states and outputs can be omitted, and targets MPC problems that are subjected to input rate and amplitude constraints only.

Algorithms that aim to solve constrained optimal control problems using first-order techniques typically require a projection onto the constraint set. In the absence of rate constraints, this projection is usually straightforward and can be computed using a closed-form formula, e.g. projection onto a box-shaped set of upper and lower actuator limits. However, if input rate (often referred to as *slew rate*) constraints are included then the projection is more complicated. We are not aware of any closed form solution to the corresponding projection problem (see Section 3.1). The simplicity of the box-projection can be recovered by introducing an augmented problem form, e.g. one that includes additional state variables (as implemented in Stellato et al. (2020) for example). While this approach is versatile in the sense that it can cope with most reasonable sets encountered in practice, the augmentation of decision variables curtails the computation speed while increasing the memory usage. It also introduces additional equality constraints to the problem, leading to difficulties in applying methods such as the fast gradient method. The question arises whether it is actually necessary to augment the decision variables in the particular case of constraints arising from input rate and amplitude constraints.

This paper suggests an approach that does *not* require augmenting the decision variables of the optimization problem. By employing a closed-form solution for a 2-dimensional rate and amplitude constraint set in combination with Dykstra's algorithm (Boyle and Dykstra, 1986), we show that the projection of a vector of arbitrary finite dimension onto the space of rate-constrained signals can be found iteratively, avoiding the need for additional state or other problem variables. Our projection algorithm is then embedded in a fast gradient method (Nesterov, 2003). Compared to the complexity introduced by the augmenta-

⋆ This research is supported by the Engineering and Physical Sciences Research Council (EPSRC) with a Diamond CASE studentship.

10.1016/j.ifacol.2020.12.070

tion of decision variables, it turns out that the additional computations required by our projection algorithm are insignificant.

This paper is organized as follows. In Section 2, we introduce both the linear MPC problem and the fast gradient method – an algorithm which is particularly suitable for solving it. Section 3 formally defines the input rate and amplitude constraint set before presenting Dykstra's method and its application to rate constraints. In Section 4, the fast gradient method is combined with Dykstra's method. The performance of our algorithm is compared with the Alternating Direction of Method of Multipliers (ADMM) for MPC problems with input rate and amplitude constraints. Both algorithms are implemented in C and tested on several examples.

2. PROBLEM STATEMENT

2.1 Model Predictive Control

Given a discrete-time linear dynamical system and an initial condition $x(t)$ at time t, a standard MPC scheme computes a control law by predicting the future evolution of the system and minimizing a quadratic objective function over some planning horizon T. This can be achieved via repeated solution of the following quadratic program (QP):

$$\min \sum_{k=0}^{T-1} x_k^{\mathrm{T}} Q x_k + u_k^{\mathrm{T}} R u_k + x_T^{\mathrm{T}} P x_T, \tag{1a}$$

$$s.t.\ x_{k+1} = Ax_k + Bu_k, \quad x_0 = x(t), \tag{1b}$$

$$(u_0, \dots, u_{T\text{-}1}) \in \mathcal{U}, \tag{1c}$$

for $k = 0, \dots, T-1$, returning at each step the optimal first input stage $u(t) = u_0^*$ as a control law. The inputs $u_k \in \mathbb{R}^{n_u}$ are constrained to the closed convex set $\mathcal{U} \subseteq \mathbb{R}^{n_u \times n_u}$. It is assumed that no constraints are imposed on the states $x_k \in \mathbb{R}^{n_x}$. The terminal cost matrix $P = P^{\mathrm{T}} \succ 0$ can be obtained from the discrete-time algebraic Riccati equation (DARE) associated to the unconstrained infinite horizon regulator problem. The QP in (1) has a unique solution if $R \succ 0$, $Q \succeq 0$ and the pairs (A, B) and $(A, Q^{\frac{1}{2}})$ are controllable and observable, respectively (Borrelli et al., 2017, Chapter 12).

By eliminating the state variables $x := (x_1, \dots, x_T)$ and defining $u := (u_0, \dots, u_{T\text{-}1})^{\mathrm{T}}$, (1) can be reformulated in its condensed form as (Borrelli et al., 2017)

$$\min\ \frac{1}{2} u^{\mathrm{T}} J u + q(x_0)^{\mathrm{T}} u, \tag{2a}$$

$$s.t.\ u \in \mathcal{U}, \tag{2b}$$

where J and $q(x_0)$, which depends linearly on x_0, arise from elimination of the equality constraints in (1b). Note that $J = J^{\mathrm{T}} \succ 0$ by the assumptions of the previous paragraph.

2.2 Fast Gradient Method

The fast gradient method belongs to the family of first-order methods that seek solutions of convex optimization problems using only the first derivative of the objective function (2a). In this paper we will use the *constant step scheme II* (Nesterov, 2003, Chapter 2.2) for strongly convex objective functions. This step scheme is known to have an *optimal* convergence rate (Nesterov, 2003, Thm. 2.2.2), but our results will hold generally for any first-order optimisation scheme employing a projection. A formulation of this algorithm is presented in (Jerez et al., 2014) and repeated below in Algorithm 1. The fixed step size β is based on the minimum and maximum eigenvalues λ_{min} and λ_{max} of $J \succ 0$, respectively.

In order to reduce the computation effort and complexity, we will not use any termination criterion for the optimisation and instead run the algorithm for a fixed number of iterations I_{max}. Based on the convergence rate results for the fast gradient method, a maximum number of iterations I_{max} can be derived that guarantees a certain level of suboptimality for all initial states $x_0 = x(t)$ within a bounded set (Richter et al., 2012).

In order to apply Algorithm 1 to our optimal control problem (2), the projection operator $\mathcal{P}_{\mathcal{U}}$ for the set $\mathcal{U}$ must be known. For a constraint set combining both input rate and amplitude constraints, this projection is not straightforward and will be addressed in Section 3.

Algorithm 1 Fast gradient method applied to (1)

Input: $x_0 = x(t)$
Output: $u(t) = u_{I_{max}}$
1: Compute $q = q(x_0)$ and set $y_1 = u_1 = 0$
2: **for** $i = 1$ to I_{max} **do**
3: $\quad t_i = (I - J/\lambda_{max}) y_i - q/\lambda_{max}$
4: $\quad u_{i+1} = \mathcal{P}_{\mathcal{U}}(t_i)$
5: $\quad y_{i+1} = (1+\beta) u_{i+1} - \beta u_i$
6: **end for**

2.3 Alternating Direction of Multipliers Method

The Alternating Direction of Method of Multipliers (ADMM) belongs to the class of augmented Lagrangian methods and is, like fast gradient method, a first-order method. ADMM algorithms are based on repeatedly minimizing the augmented Lagrange function w.r.t. the primal variables and maximizing the same function w.r.t. to the dual variables (Boyd and Vandenberghe, 2004, Chapter 5). Assuming that the input constraint set (2b) can be represented as a polyhedron, i.e.

$$\mathcal{U} = \{u \in \mathbb{R}^{T\text{-}1} | \underline{v} \le Ku \le \bar{v}\} \tag{3}$$

the optimization problem (2) can be reformulated as

$$\min\ \frac{1}{2} u^{\mathrm{T}} J u + q(x_0)^{\mathrm{T}} u, \tag{4a}$$

$$s.t.\ Ku - v = 0, \tag{4b}$$

$$\underline{v} \le v \le \bar{v}, \tag{4c}$$

where the constraint variables $v \in \mathbb{R}^{n_v}$ and equality constraints (4b) were introduced. The augmented Lagrangian for (4) can be written as

$$\begin{aligned} L(u, v, \gamma) = &\frac{1}{2} u^{\mathrm{T}} J u + q(x_0)^{\mathrm{T}} u + \frac{\rho}{2} \|Ku - v\|_2^2 \\ &+ \gamma^{\mathrm{T}} (Ku - v) + \mathcal{I}_{[\underline{v}, \bar{v}]}(v), \end{aligned} \tag{5}$$

where $\mathcal{I}_{[\underline{v}, \bar{v}]}$ is the indicator function for the set $\mathcal{V} = \{v \mid \underline{v} \le v \le \bar{v}\}$ and the penalty parameter $\rho > 0$ and the dual variables γ are associated with the constraint (4b). A standard ADMM scheme solves (4) by repeatedly minimizing (5) w.r.t. u and v and updating the dual variables γ using an approximate gradient ascent method. The algorithm is summarized in Algorithm 2, where the

saturation function $\mathrm{sat}_{[\underline{v},\bar{v}]}(\cdot)$ is used, which limits its argument to $\underline{v}$ and $\bar{v}$. Reformulation (4) simplifies the projection involved in the algorithm: instead of projecting onto the polyhedron $\mathcal{U}$, which might be as hard as solving (2), the projection onto $\mathcal{V}$ is given by the saturation function. For the purposes of this paper, the critical distinction between our optimisation problem in the form (2) (and its solution via fast gradient method) and (4) (and its solution via ADMM) is that the form (2) has a positive definite J and does not allow for equality constraints. The problem form (4) makes neither restriction.

Algorithm 2 ADMM applied to (1) with set (3)

Input: $x_0 = x(t)$
Output: $u(t) = u_{I_{max}}$
1: Compute $q = q(x_0)$ and set $\gamma_0 = 0$
2: **for** $i = 1$ to I_{max} **do**
3: Solve for u_i: $\left(J + \rho K^{\mathrm{T}} K\right) u_i = K^{\mathrm{T}}(\rho v_{i\text{-}1} - \gamma_{i\text{-}1}) - q$
4: $v_i = \mathrm{sat}_{[\underline{v},\bar{v}]}\left\{K u_i + \rho^{-1}\gamma_{i\text{-}1}\right\}$
5: $\gamma_i = \gamma_{i\text{-}1} + \rho(K u_i - v_i)$
6: **end for**

3. INPUT CONSTRAINT PROJECTION METHOD

Given a nonempty closed convex set $\mathcal{X} \subseteq \mathbb{R}^{N\times N}$, the Euclidean projection x^* of a point $x^\circ \in \mathbb{R}^N$ is defined as the minimizer of the following optimization problem:

$$x^* = \underset{x\in\mathcal{X}}{\arg\min} \|x - x^\circ\|_2 =: \mathcal{P}_{\mathcal{X}}(x^\circ). \tag{6}$$

By the assumptions on the set $\mathcal{X}$, the optimization problem (6) admits a unique solution (Bauschke and Combettes, 2011, Chapter 3.2).

3.1 Rate and Amplitude Constraint Set

Given a maximum allowable input amplitude $a \in \mathbb{R}^{n_u} > 0$ and rate $r \in \mathbb{R}^{n_u} > 0$, define the amplitude constraint set as $\mathcal{A}_k := \left\{u \in \mathbb{R}^{n_u(T-1)} \,|\, |u_k| \le a\right\}$ and the rate constraint set as $\mathcal{R}_k := \left\{u \in \mathbb{R}^{n_u(T-1)} \,|\, |u_k - u_{k\text{-}1}| \le r\right\}$ for $k = 0, \ldots, T-1$, where the inequalities are applied elementwise. The set $\mathcal{R}_0$ includes the input $u_{\text{-}1}$ that is treated as a fixed constant stemming from the actual input of the system at time $t-1$. We exclude the trivial case where $\mathcal{A}_k$ is entirely contained in $\mathcal{R}_k$ by assuming that $0 \le r \le 2a$ for all n_u elements. The input rate and amplitude constraint set for problem (2) is obtained as the intersection of $\mathcal{A} = \mathcal{A}_0 \cap \cdots \cap \mathcal{A}_{T\text{-}1}$ and $\mathcal{R} = \mathcal{R}_0 \cap \cdots \cap \mathcal{R}_{T\text{-}1}$, i.e.

$$\mathcal{U} := \{u \in \mathbb{R}^{n_u(T\text{-}1)} \big| |u_k| \le a, |u_k - u_{k\text{-}1}| \le r \,\forall\, k \in \mathcal{K}\}. \tag{7}$$

Because the constraints are not coupled among the elements of $u_k \in \mathbb{R}^{n_u}$, we will assume throughout that $n_u = 1$ for clarity of exposition. However, all of our results apply in the case that $n_u > 1$.

While the projection onto $\mathcal{A}$ is given by saturating the elements of u to $\pm a$, we know of no tractable closed-form solution for $\mathcal{P}_{\mathcal{R}}$ (see also Bauschke and Koch (2013)) and hence also not for $\mathcal{P}_{\mathcal{U}}$. In order to see why this projection is not straightforward, $\mathcal{P}_{\mathcal{R}}(u^\circ)$ can be redefined as a dynamic programming problem (Kempf et al., 2020). It quickly becomes apparent that the evaluation of $3^{T\text{-}1}$ conditions is required for obtaining $\mathcal{P}_{\mathcal{R}}(u^\circ)$. While it can be possible to derive a formula for small horizons, the solution for dynamic programming becomes intractable for larger horizons.

Another approach to obtain $\mathcal{P}_{\mathcal{R}}(u^\circ)$ or directly $\mathcal{P}_{\mathcal{U}}(u^\circ)$ would be to define a multi-parametric program (Borrelli et al., 2017, Chapter 2) with parameters u° and $u_{\text{-}1}$. The multi-parametric solution of the projection results in a piecewise affine function (PWA) of the parameters $u_{\text{-}1}$ and u°, i.e. n affine functions defined on n disjoint sets. While an explicit solution could be computed using dedicated software, e.g. (Herceg et al., 2013), it is expected that the number of regions n would be at least $3^{T\text{-}1}$, which only accounts for projecting onto $\mathcal{R}$.

3.2 2-Dimensional Projection

Consider the input rate and amplitude constraint set (7) for $n_u = 1$ and $T = 2$, i.e. $u = (u_0, u_1)^{\mathrm{T}}$, which can be represented as

$$\mathcal{U}_1 := \{u \in \mathbb{R}^{n_u(T\text{-}1)} \,\big|\, \underline{a} \le u_0 \le \bar{a},\ |u_1| \le a,\ |u_1 - u_0| \le r\}, \tag{8}$$

where $\underline{a} := \max(-a, -r + u_{\text{-}1})$ and $\bar{a} := \min(a, r + u_{\text{-}1})$. When r and a are fixed, the shape of set (7) depends on parameter $u_{\text{-}1}$ and resembles a rectangle with cut-off top-left and bottom-right corners. Each facet and corner of this set define disjoint regions for which a specific projection function is valid (Kempf et al., 2020). If these regions and functions are parametrized by $u_{\text{-}1}$, the projection consists of two steps: Firstly, determine to which region the point belongs. Secondly, apply the projection function of that particular region. In other words, we identify explicitly the PWA solution to the problem of projection onto the set $\mathcal{U}_1$ following the general method of (Borrelli et al., 2017, Chapter 2). A complete C-language implementation of the 2-dimensional projection can be found in (Kempf, 2019a).

3.3 Dykstra's Algorithm

Dykstra's algorithm (Boyle and Dykstra, 1986) was first published in 1983 as an extension to von Neumann's *Alternating Projections Method* (Von Neumann, 1951), which is suitable for finding a point lying in the intersection $\mathcal{U} = \mathcal{U}_1 \cap \cdots \cap \mathcal{U}_N$ of N closed convex sets $\mathcal{U}_i$ by cyclically projecting onto the sets $\mathcal{U}_i$. While von Neumann's algorithm only finds *some* point in $\mathcal{U}$, Dykstra's algorithm determines the Euclidean projection $x^* = \mathcal{P}_{\mathcal{U}}(x^\circ)$ of x° onto $\mathcal{U}$. Both algorithms circumvent the potentially complicated projection $\mathcal{P}_{\mathcal{U}}$ by iteratively applying the (known) projections $\mathcal{P}_{\mathcal{U}_i}$. The method is summarized below in Algorithm 3 for the case that $\mathcal{U} = \mathcal{U}_1 \cap \mathcal{U}_2$.

Algorithm 3 Dykstra's Algorithm for two sets

Input: x°
Output: $\mathcal{P}_{\mathcal{U}}(x^\circ) = x_{i+1}$
1: Set $x_0 = x^\circ$, $\mu_0 = 0$ and $\gamma_0 = 0$
2: **for** $i = 1$ to I_{max} **do**
3: $y_i = \mathcal{P}_{\mathcal{U}_1}(x_i + \mu_i)$
4: $\mu_{i+1} = \mu_i + x_i - y_i$
5: $x_{i+1} = \mathcal{P}_{\mathcal{U}_2}(y_i + \gamma_i)$
6: $\gamma_{i+1} = \gamma_i + y_i - x_{i+1}$
7: **if** $\|x_{i+1} - x_i\|_\infty < \epsilon$ **then break**
8: **end for**

What distinguishes Dykstra's algorithm from von Neumann's is the choice of variables μ^k and γ^k that track

the residuals from projecting onto $\mathcal{U}_1$ and $\mathcal{U}_2$ (Tibshirani, 2017). It can be shown that Dykstra's algorithm is equivalent to ADMM applied to problem (6) for the case that $\mathcal{U} = \mathcal{U}_1 \cap \mathcal{U}_2$ (Tibshirani, 2017). In addition, it has been proved that Algorithm 3 always converges to the Euclidean projection onto $\mathcal{U}$, provided that the sets $\mathcal{U}_1$ and $\mathcal{U}_2$ are closed convex sets and their intersection is nonempty (Boyle and Dykstra, 1986; Bauschke and Combettes, 2011).

Before applying Algorithm 3 to the rate and amplitude constraint set (7), it remains to show that the set (7) can be formulated as the intersection of two closed convex sets with known projection operators. In Section 3.2 we demonstrated how to project onto (7) for $T = 2$ using a geometrical approach. Set (7) can be represented as the intersection of $N = T - 2$ closed convex sets $\mathcal{U}_k$, where $\mathcal{U}_1$ is defined in (8) and

$$\mathcal{U}_k := \{u \in \mathbb{R}^{n_u(T\text{-}1)} \,|\, |u_k| \leq a, |u_{k\text{-}1}| \leq a, |u_k - u_{k\text{-}1}| \leq r\}, \quad (9)$$

for $k = 2, \ldots, T-1$. Let π_k denote the procedure presented in Section 3.2 applied element-wise to the n_u elements of $u_{k\text{-}1}$ and u_k, respectively, with $\bar{a} = -\underline{a} = a$ for $k > 1$. Let π_k^1 and π_k^2 be the resulting n_u projections of $u_{k\text{-}1}$ and u_k, respectively. Then $\mathcal{P}_{\mathcal{U}_k} \in \mathbb{R}^{n_u(T\text{-}1)}$ can be written as

$$\mathcal{P}_{\mathcal{U}_k}(u) = (u_0, u_1, \ldots, \pi_k^1, \pi_k^2, \ldots, u_{T\text{-}1})^{\mathrm{T}}, \quad (10)$$

where elements $u_{k\text{-}1}$ and u_k have been replaced by π_k^1 and π_k^2, respectively. Assume for the purpose of explanation that T is even and let $\mathcal{U}_e$ and $\mathcal{U}_o$ denote the intersection of sets $\mathcal{U}_i$ grouped by even and odd indices, respectively. The projections $\mathcal{P}_{\mathcal{U}_e}(u^\circ) \in \mathbb{R}^{n_u(T\text{-}1)}$ and $\mathcal{P}_{\mathcal{U}_o}(u^\circ) \in \mathbb{R}^{n_u(T\text{-}1)}$ are obtained by combining the corresponding projections from (10) and given by

$$\mathcal{P}_{\mathcal{U}_e}(u) = (u_0, \pi_2^1, \pi_2^2, \pi_4^1, \ldots, \pi_{T\text{-}2}^2, u_{T\text{-}1})^{\mathrm{T}}, \quad (11a)$$

$$\mathcal{P}_{\mathcal{U}_o}(u) = (\pi_1^1, \pi_1^2, \pi_3^1, \pi_3^2, \ldots, \pi_{T\text{-}1}^1, \pi_{T\text{-}1}^2)^{\mathrm{T}}. \quad (11b)$$

By setting $\mathcal{P}_{\mathcal{U}_1} = \mathcal{P}_{\mathcal{U}_e}$ and $\mathcal{P}_{\mathcal{U}_2} = \mathcal{P}_{\mathcal{U}_o}$, Algorithm 3 can be applied to project onto the input rate and amplitude constraint set (7). The choice of using a 2-dimensional projection in combination with Dykstra's method is mainly motivated by the geometrical approach of Section 3.2. Another possibility would be to compute a formula for a 3-dimensional projection $\tilde{\pi}_k$ onto the set $\mathcal{U}_k \cap \mathcal{U}_{k+1}$ for $n_u = 1$ using one of the methods outlined in Section 3.1. Compared to the 2-dimensional case, fewer $\tilde{\pi}_k$ would have to be evaluated at the expense of increased complexity. How this would affect the computational performance and the convergence rate of the method is not clear a-priori (Boyle and Dykstra, 1986).

Figure 1 compares the output of Algorithm 3 applied to the input rate and amplitude constraint set (7) with $r = a = 1$ for different horizons T. The distance of iterates x_{i+1} from Algorithm 3 to the solution $x^* = \mathcal{P}_{\mathcal{U}}(x^\circ)$ obtained using an interior-point method is shown. For the figure we selected 100 starting points x° from a normal distribution with zero mean and a standard deviation of 100 (left) and 10 (right), respectively.

4. MPC FOR SYSTEMS WITH INPUT RATE AND AMPLITUDE CONSTRAINTS

By replacing the projection $\mathcal{P}_{\mathcal{U}}$ with Dykstra's algorithm, the fast gradient method (Algorithm 1) and Dykstra's projection method (Algorithm 3) are combined (Algorithm 1&3). Compared to the ADMM implementation, Algorithm 1&3 bears several advantages. Dykstra's algorithm makes the augmentation of decision variables superfluous. While the ADMM formulation (4) requires $2T - 1$ decision variables, Algorithm 1&3 reduces the number of decision variables to T. This not only greatly reduces the computation time, as the next section will show, but also lowers the memory footprint. If we assume that all matrices are dense and neglect the storage of vectors, then Algorithm 1&3 reduces the memory footprint by approximately $T^2/(T^2 + (T-1)^2 n_v/n_u)$, which roughly amounts to a reduction of 50% for set (7) and large T. Dykstra's algorithm barely introduces any memory footprint because it only involves Boolean operations and vector additions. Moreover, arrays allocated by Algorithm 1 can be used as temporary placeholders to execute Dykstra's method. Stand-alone C-language implementations of Algorithm 1 and 3 can be found under (Kempf, 2019a,b). To this end, note that Algorithms 1, 2 and 1&3 could be warm-started using the solution computed at time step $t-1$, e.g. setting $y_1 = u_{\text{-}1} = u(t-1)$ in Algorithm 1&3.

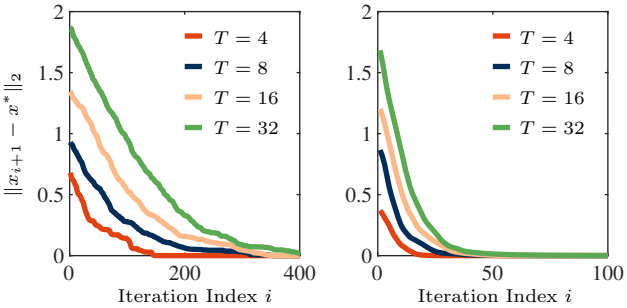

Fig. 1. Application of Algorithm 3 to (7) with $r = a = 1$. Shown is the distance of x_{i+1} to an accurate projection $x^* = \mathcal{P}_{\mathcal{U}}(x^\circ)$ for distant (left) and close (right) starting points x°, respectively.

4.1 Numerical Studies

Algorithm 1&3 is compared against an ADMM implementation which – as described in Section 2.3 – commonly uses an augmentation of decision variables to simplify the projection. While Algorithm 1&3 was implemented in C and can be found under Kempf (2019a,b), the Operator Splitting Quadratic Program solver (Stellato et al., 2020) – a constrained QP solver that uses ADMM – was employed for Algorithm 2. In order to avoid refactoring the matrix on the left-hand side of step 3 of Algorithm 2, our benchmark ADMM implementation uses a constant penalty parameter ρ. Neither of the C-programs includes a non-standard C-library.

Figure 2 compares the average execution times of one iteration of the C-language implementation of the ADMM and the fast gradient method applied to problem (2) with the input rate and amplitude constraint set $\mathcal{U}$ as defined in (7) with $r = a = 1$. The OSQP solver uses a matrix-factorization to solve step 3 of Algorithm 2 and the time for factorizing the matrix is excluded from Figure 2. The problem data (J, q) is randomly generated and the average execution times are benchmarked over 100 problems per horizon. Figure 2 reveals that the combination of the fast gradient method and Dykstra's method greatly reduces the computation time for one solver iteration. The perfor-

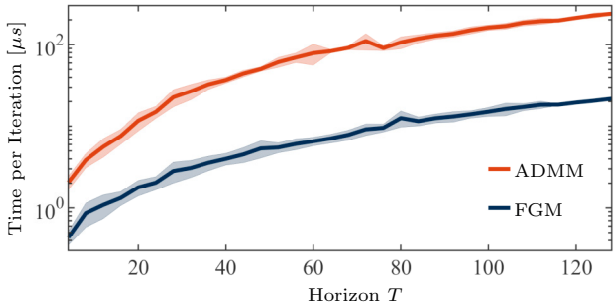

Fig. 2. Average execution times and standard deviations (shaded) of one iteration of Algorithm 2 (ADMM, red) and Algorithm 1&3 (FGM, blue), respectively, applied to problem (2) with set (7).

mance gain is due to the fact that Dykstra's method makes the augmentation of decision variables (4) unnecessary, which in case of the input rate and amplitude set amounts to tripling the number of decision variables. Because Dykstra's method involves only vector additions and Boolean operations, the projection algorithm only requires a few processor cycles. The termination criterion for Dykstra's algorithm is checked on every 10th iteration.

Figure 3 compares the practical convergence behaviour of Algorithm 2 (ADMM, red) and Algorithm 1&3 (FGM, blue) applied to problem (2) with the input rate and amplitude constraint set as defined in (7) with $r = a = 1$. The average distance between a high-accuracy solution u^* calculated using an interior-point method and the solution at iteration i of Algorithm 2 and 1&3, respectively, are shown. The problem data (J, q) is randomly generated and the distances are averaged over 100 problems per horizon T. For Algorithm 1&3, Dykstra's method is implemented as follows: An initial verification is applied to avoid executing the algorithm for vectors t_i that lie inside set (7). If $t_i \notin \mathcal{U}$, Dykstra's method is run for a fixed number of iterations $J_{max} = 50$.

4.2 Example

We study the application of Algorithms 2 and 1&3 to an aircraft stabilization problem. The linearized model is taken from (Kose and Jabbari, 2001), discretized with a sampling time of $T_s = 1ms$ and an MPC problem formulated in its condensed form (2) for horizons $T = \{4, 8, 16, 32\}$ and with $Q = R = I$. The aircraft model features $n_u = 3$ inputs, i.e. $u_k = (u_k^{ail}, u_k^{stab}, u_k^{rud})^{\mathrm{T}}$, where u_k^{ail}, u_k^{stab} and u_k^{rud} denote the differential aileron deflection, the differential stabilizer deflection and the rudder deflection, respectively. The inputs are constrained by rate and amplitude constraints with $(a^{ail}, a^{stab}, a^{rud}) = (25, 24, 30)$ and $(r^{ail}, r^{stab}, r^{rud}) = (200T_s, 80T_s, 82T_s)$, respectively. While Algorithms 2 and 1&3 are used to solve the condensed MPC problem (2), the dynamics of the aircraft are simulated using (1b) starting with a random non-zero initial condition. The closed-loop simulation is run for 1000 time steps. In order to be able to compare the solution accuracies and avoid computing the number of iterations I_{max} required for a certain level of optimality (see Section 2.2), a termination criterion for Algorithm 1&3 is introduced:

$$r_i < \epsilon_{abs} + \epsilon_{rel} r_{i\text{-}1}, \qquad r_i := \|u_{i+1} - u_i\|_\infty. \tag{12}$$

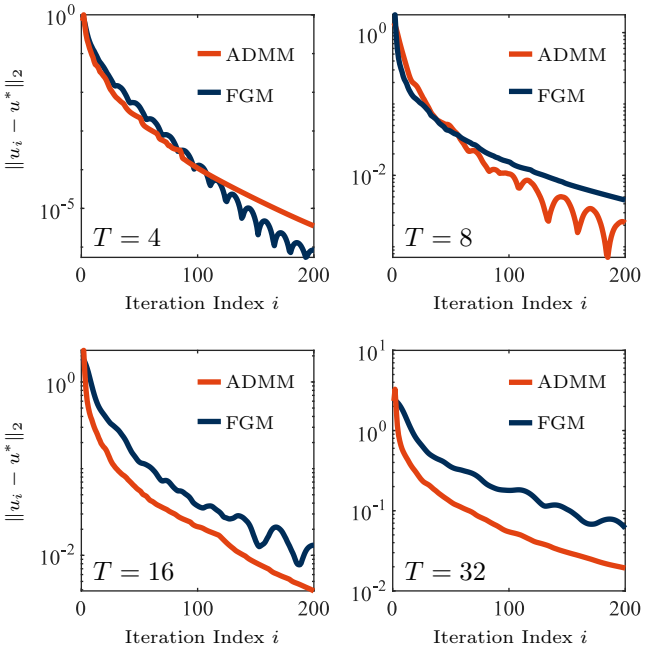

Fig. 3. Practical convergence behaviour of Algorithm 2 (ADMM, red) and Algorithm 1&3 (FGM, blue) applied to problem (2) with set (7) for $T = \{4, 8, 16, 32\}$. Shown is the distance between the iterates and a solution u^* computed using an interior-point method. Algorithm 2 uses a constant penalty parameter ρ. The problem data is randomly generated and the distances are averaged over 100 problems.

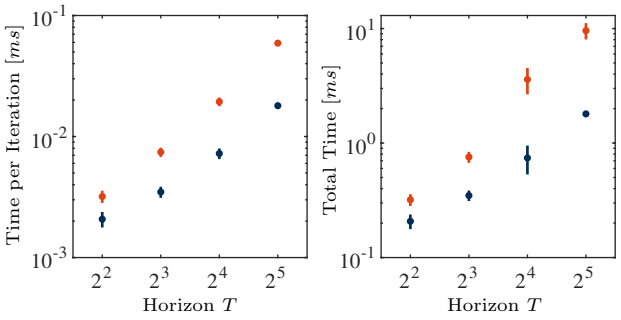

Fig. 4. Comparison of the execution times of Algorithm 2 (red) and Algorithm 1&3 (blue), respectively, applied to the aircraft stabilization problem of Section 4.2. The vertical bars indicate the standard deviations.

Both algorithms check for termination on every 10th iteration and its is verified that they produce a similar closed-loop behavior.

Figure 4 depicts the average time per iteration (top) and the average total time (bottom) required to solve one instance of problem (1). Both rows are averaged over the 1000 time steps of the closed-loop simulation. Compared to Figure 2 where $n_u = 1$ was used, it can be seen that the $n_u = 3$ inputs introduce an overhead and slow down Algorithm 1&3 for smaller horizons. The performance advantage is approximately regained for $T = 32$. The bottom row confirms the results from Figure 3, which showed that a similar convergence behavior can be expected from Algorithms 2 and 1&3.

5. CONCLUSIONS AND FUTURE WORK

This paper demonstrated the use of Dykstra's method to project onto the input rate and amplitude constraint sets. A procedure to solve the 2-dimensional projection was presented and a C-language implementation provided. It was shown how the input constraint set can be represented as the intersection of two sets. Using the formula for the 2-dimensional case, the projection was extended to higher dimensions and computed iteratively using Dykstra's method. Several simulations showed the fast convergence of the projection algorithm. The choice of using an underlying 2-dimensional projection was motivated by a geometrical approach. It is certain that choosing a different dimension for the underlying projection would affect the performance of the algorithm. Investigating the performance for different underlying projections is subject of future work.

In order to solve a linear MPC problem constrained by input rate and amplitude limits, Dykstra's method was embedded in an fast gradient method that uses a constant step scheme. The resulting algorithm was compared against an ADMM implementation. The ADMM uses a decision variable augmentation to accommodate the constraints. Using a C-language implementation applied to several example problems, it was shown that the combination of the fast gradient method and Dykstra's algorithm significantly reduces the computation time compared to an ADMM implementation. Moreover, the practical convergence behaviour was simulated and it was shown that the combination of fast gradient method and Dykstra's method converges as quickly as the ADMM implementation. In addition, it was demonstrated that the combined algorithms approximately halve the memory footprint, which is of particular importance for an implementation on an embedded system.

While it has been shown that the practical convergence behaviour of Algorithm 1&3 matches the one of Algorithm 2, future work will focus on establishing convergence criteria for the combination of Algorithms 1 and 3. Dykstra's method introduces a projection error that can be made arbitrarily small by increasing the number of iterations (Bauschke and Combettes, 2011, Chapter 29.1). The convergence of ADMM under an inexact projection has been proved in Eckstein and Bertsekas (1992, Theorem 8), where it was shown that the algorithm converges provided that the sum of the projection errors remains bounded. Since the projection error of Dykstra's algorithm can be made arbitrarily small, the boundedness of the sum of the projection errors can be enforced and it might be expected that a similar proof can be formulated for Algorithm 1&3. A practically relevant situation is when the fast gradient method as well as Dykstra's method are run for a fixed number of iterations. A proof exists for the fast gradient method with an exact projection in Richter et al. (2012), where it was shown that a certain level of suboptimality can be guaranteed provided that the initial condition x_0 of the MPC problem (1) lies within a bounded set. Future efforts will also aim at extending the proof from Richter et al. (2012) to the case of a projection using Dykstra's method.

REFERENCES

Bauschke, H. and Combettes, P. (2011). *Convex Analysis and Monotone Operator Theory in Hilbert Space*. Springer.

Bauschke, H. and Koch, V. (2013). Projection methods: Swiss army knives for solving feasibility and best approximation problems with halfspaces. *ArXiv e-prints*.

Borrelli, F., Bemporad, A., and Morari, M. (2017). *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

Boyle, J.P. and Dykstra, R.L. (1986). A method for finding projections onto the intersection of convex sets in Hilbert spaces. In *Advances in Order Restricted Statistical Inference*, 28–47. Springer New York.

Eckstein, J. and Bertsekas, D.P. (1992). On the Douglas–Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55(1), 293–318.

Herceg, M., Kvasnica, M., Jones, C., and Morari, M. (2013). Multi-Parametric Toolbox 3.0. In *2013 12th European Control Conference (ECC)*, 502–510. Zürich, Switzerland. URL http://control.ee.ethz.ch/~mpt.

Jerez, J.L., Goulart, P.J., Richter, S., Constantinides, G.A., Kerrigan, E.C., and Morari, M. (2014). Embedded online optimization for model predictive control at megahertz rates. *IEEE Transactions on Automatic Control*, 59(12), 3238–3251.

Kempf, I. (2019a). Box-rate-projection. URL https://github.com/kmpape/box_rate_projection.

Kempf, I. (2019b). Fast-gradient-method. URL https://github.com/kmpape/fast_gradient_method.

Kempf, I., Goulart, P., and Duncan, S. (2020). Fast gradient method for model predictive control with input rate and amplitude constraints. *ArXiv e-prints*.

Kose, I.E. and Jabbari, F. (2001). Control of systems with actuator amplitude and rate constraints. In *Proceedings of the 2001 American Control Conference*, volume 6, 4914–4919.

Nesterov, Y. (2003). *Introductory Lectures on Convex Optimization: A Basic Course*. Applied Optimization. Springer Science & Business Media.

Richter, S., Jones, C.N., and Morari, M. (2012). Computational complexity certification for real-time mpc with input constraints based on the fast gradient method. *IEEE Transactions on Automatic Control*, 57(6), 1391–1403.

Saberi, A., Stoorvogel, A., and Sannuti, P. (1999). *Control of Linear Systems with Regulation and Input Constraints*. Communications and Control Engineering. Springer London.

Stellato, B., Banjac, G., Goulart, P.J., Bemporad, A., and Boyd, S. (2020). OSQP: An operator splitting solver for quadratic programs. *Mathematical Programming Computation*.

Tibshirani, R.J. (2017). Dykstra's algorithm, ADMM, and coordinate descent: Connections, insights, and extensions. In *Advances in Neural Information Processing Systems 30*, 517–528. Curran Associates, Inc.

Von Neumann, J. (1951). *Functional Operators: The Geometry of Orthogonal Spaces*. Number Bd. 2 in Annals of Mathematics Studies. Princeton University Press.