Real-Time Input-Constrained MPC Using Fast Gradient Methods

Stefan Richter, Colin N. Jones and Manfred Morari

Abstract— **Linear quadratic model predictive control (MPC) with input constraints leads to an optimization problem that has to be solved at every instant in time. Although there exists computational complexity analysis for current online optimization methods dedicated to MPC, the worst case complexity bound is either hard to compute or far off from the practically observed bound. In this paper we introduce fast gradient methods that allow one to compute *a priori* the worst case bound required to find a solution with pre-specified accuracy. Both warm- and cold-starting techniques are analyzed and an illustrative example confirms that small, practical bounds can be obtained that together with the algorithmic and numerical simplicity of fast gradient methods allow online optimization at high rates.**

I. Introduction

This paper considers the computational aspects of model predictive control (MPC) of discrete-time, linear systems with quadratic costs and linear inequality constraints on the inputs. In MPC one aims to minimize a specified cost over a horizon forward in time and applies the first control of the obtained sequence of controls to the plant. This scheme is known as *Receding Horizon Control* and is repeated at every time-step given the new state information. Since the solution of an optimization problem is required at every time-step, the first applications of MPC were restricted to systems with slow dynamics, enabling large sampling intervals and therefore sufficient time to solve the optimization problem.

In recent years, MPC has made its way into control applications with short sampling intervals, mainly because computational power has increased and new techniques to solve the optimization problem have emerged. One such technique is multi-parametric programming, which allows one to pre-compute the solution for every state offline [1]. The explicit solution is a piece-wise affine map over a polyhedral partition of the state-space and can be stored efficiently such that fast online look-up is ensured. As the number of polyhedral partitions grows exponentially in the size of the MPC problem in the worst case, recent approximation methods reduce complexity while maintaining stability and feasibility of the obtained control law, see e.g. [2]. However, approximate explicit solutions are also subject to worst case exponential growth limiting explicit MPC to small and medium-sized control problems.

Online solution methods in MPC are generally used for larger scale problems (e.g. > 10 state dimensions) that in the considered context divide into two categories of iterative solution schemes: Active set methods and interior point methods (see e.g. [3] for details). For active set methods convergence can only be guaranteed after a finite number of steps in general. Nevertheless, these methods show good practical performance. A dedicated implementation of an active set method for MPC is discussed in [4]. It makes use of the piece-wise affine structure of the explicit MPC solution as well as re-uses the solution from the previous time-step, a strategy known as warm-starting. Interior point methods allow one to bound the number of iterations, however, the obtained bound is too conservative and is known to be much larger than the number of iterations seen in practice [5, §11]. A fast implementation of such a method is reported in [6] and its applicability is demonstrated in simulation studies. The cut down of computational time is achieved there by exploiting the special structure of the MPC problem as well as by applying warm-starting and early stopping of the iterative scheme. For both approaches neither feasibility nor stability can be guaranteed under a specified fixed run-time for the general case of state and input constraints.

More recently, the combination of an approximate explicit solution and an active set method for MPC with linear cost was introduced by [7] where it is shown how performance and run-time guarantees can be obtained *a priori*. Again, the explicit part limits its application to medium-sized problems.

In this paper, we propose to use the iterative algorithmic scheme of fast gradient methods developed in [8] to solve the optimization problem for the specific case of MPC with input constraints. We show that fast gradient methods, apart from allowing an intriguing simple implementation, provide the possibility to obtain a practical upper bound on the number of iterations required to obtain a solution of pre-specified accuracy. Both warm- and cold-starting strategies are discussed and corresponding upper bounds are derived. The upper bounds are generally computed by multi-level programming, but for the case of cold-starting turn out to be easily found. In the case of warm-starting we present a way of interpreting a central entity in the upper bound in terms of optimal control suggesting that the effort required to solve the optimization problem does not necessarily grow with the control horizon. An illustrative example backs up the theoretical results.

Note that for the class of input-constrained control problems a variety of other control approaches that complement MPC exists in literature (see e.g. [9]).

II. MPC Control Problem Formulation

We consider discrete-time, linear, time-invariant systems

$$x_{k+1} = Ax_k + Bu_k \quad , \quad \forall k = 0, 1 \dots \quad , \tag{1}$$

S. Richter, C.N. Jones and M. Morari are with the Department of Electrical Engineering, Swiss Federal Institute of Technology Zürich, ETL I28, Physikstrasse 3, 8092 Zürich, Switzerland, e-mail: richters|cjones|morari@ee.ethz.ch

where $x_k \in \mathbb{R}^n$ and $u_k \in \mathbb{R}^m$ denote the state and control input at time-step k respectively. We restrict the control input to belong to a convex, compact set described by the intersection of a finite number of halfspaces (*polytope*), i.e. $u_k \in \mathbb{U} \subset \mathbb{R}^m$, where we assume the origin to be contained in its interior.

The regulator MPC problem

$$\begin{aligned} J_N^*(x) := \min \frac{1}{2} x_N^T P x_N + \frac{1}{2} \sum_{k=0}^{N-1} x_k^T Q x_k + u_k^T R u_k \qquad (2) \\ \text{subject to } x_{k+1} = Ax_k + Bu_k \ , \quad \forall k = 0 \dots N-1 \\ u_k \in \mathbb{U} \ , \quad \forall k = 0 \dots N-1 \\ x_0 = x \ , \end{aligned}$$

is solved at every time-step for the current state x, where N denotes the finite control horizon, matrix $Q \in \mathbb{R}^{n \times n}$ is positive semi-definite, i.e. $Q \geq 0$, and matrix $R \in \mathbb{R}^{m \times m}$ is positive definite, i.e. $R > 0$. We assume system (1) to be stable such that the positive definite terminal penalty matrix $P \in \mathbb{R}^{n \times n}$ follows from the Lyapunov Equation $A^T P A + Q = P$. With respect to this definition of matrix P, the term $1/2 x_N^T P x_N$ in (2) summarizes the infinite horizon cost when no control is applied at state x_N [10].

Remark 1: The fast gradient scheme in this paper also works for unstable systems (1) but additional measures have to be employed to ensure stability without introducing a terminal region, e.g. as described in [10, §3.7.4.1].

If all states in (2) are expressed as a linear function of the initial state x and the inputs, problem (2) can be rewritten as

$$\begin{aligned} J_N^*(x) = \min \ J_N(U;x) \qquad (3) \\ \text{subject to } U \in \mathbb{U}^N \ , \end{aligned}$$

where the sequence of inputs over the control horizon is defined as $U = (u_0, u_1, \dots, u_{N-1})$ and constrained to be in set $\mathbb{U}^N \subset \mathbb{R}^{Nm}$, being the direct product of N input sets $\mathbb{U}$. The objective function in (3) is

$$J_N(U;x) = \frac{1}{2} U^T \mathscr{T} U + x^T \mathscr{L} U + \frac{1}{2} x^T \mathscr{M} x \qquad (4)$$

where Hessian matrix $\mathscr{T} \in \mathbb{R}^{Nm \times Nm}$ is positive definite and matrices $\mathscr{L} \in \mathbb{R}^{n \times Nm}$, $\mathscr{M} \in \mathbb{R}^{n \times n}$ are easily derived from (2), cf. [1].

III. Gradient Methods for Convex Optimization

This section introduces both traditional gradient methods and fast gradient methods, which appear to be less well-known in the control community. We cover the main algorithmic schemes and introduce the generalization to constrained optimization problems. At the end of this section, the main convergence results for both gradient methods are discussed. Note that the content of this section closely follows [11, §2].

Throughout this section we consider the following optimization problem:

$$\begin{aligned} f^* = \min f(z) \qquad (5) \\ \text{subject to } z \in \mathbb{C} \ , \end{aligned}$$

where $\mathbb{C} \subseteq \mathbb{R}^p$ is a closed, convex set and f is a real-valued, convex function $f : \mathbb{R}^p \to \mathbb{R}$, which is at least once continuously differentiable on $\mathbb{C}$, such that it is lower-bounded by

$$f(z) \geq f(y) + \nabla f(y)^T (z-y) \ , \quad \forall y, z \in \mathbb{C} \ . \qquad (6)$$

In convergence analysis of optimization methods additional conditions on the function f are often useful and one will see that they are satisfied for MPC problem (3). One such additional condition is Lipschitz continuity of the gradient.

Definition 1 (Lipschitz Continuity of Gradient): The gradient of a continuously differentiable function f is Lipschitz continuous on set $\mathbb{C}$ whenever there exists a Lipschitz constant $L \geq 0$ such that

$$\|\nabla f(z) - \nabla f(y)\| \leq L \|z-y\| \ , \quad \forall z, y \in \mathbb{C} \ ,$$

where $\|.\|$ denotes the Euclidean norm throughout the paper. Lipschitz constant L allows for upper-bounding f by

$$f(z) \leq f(y) + \nabla f(y)^T (z-y) + \frac{L}{2} \|z-y\|^2 \ , \ \forall z, y \in \mathbb{C} \ . \ (7)$$

Furthermore, we assume function f to be strongly convex.

Definition 2 (Strong Convexity): A continuously differentiable function f is called strongly convex on set $\mathbb{C}$ if there exists a convexity parameter $\mu > 0$ such that

$$f(z) \geq f(y) + \nabla f(y)^T (z-y) + \frac{\mu}{2} \|z-y\|^2 \ , \ \forall z, y \in \mathbb{C} \ . \ (8)$$

Strong convexity implies that function f can be lower-bounded by a quadratic. Comparing (8) with the quadratic upper bound in (7) the general relation $L \geq \mu$ gets intuitively clear.

Note that strongly convex functions f with Lipschitz continuous gradient are regarded as the best possible objectives when it comes to solving (5) and we will see in Section IV that the objective function in MPC problem (3) belongs to this class.

A. Traditional Gradient Methods

In the case of unconstrained optimization, i.e. $\mathbb{C} = \mathbb{R}^p$, a traditional gradient method will find a new iterate z_{i+1} by following the anti-gradient $-h_i \nabla f(z_i)$ at the previous iterate z_i, i.e. $z_{i+1} = z_i - h_i \nabla f(z_i)$. The factor $h_i > 0$ is called the step size and ensures that the iterative scheme forms a so-called relaxation sequence $\{f(z_i)\}$ whereby

$$f(z_{i+1}) \leq f(z_i) \ , \quad \forall i \geq 0 \ . \qquad (9)$$

Several step size rules exist in the literature. The simplest of them, and also the choice in this paper, is a constant step size $h_i = h > 0 \ , \ \forall i \geq 1$ that is defined *a priori*. This strategy is often applied if the function is convex since it can be proven to give the same rate of convergence as other step size rules (see [11, §2.1.5] and references therein).

Interestingly, the same algorithmic scheme as before can be applied for constrained optimization problems too, where $\mathbb{C} \subset \mathbb{R}^p$, the only difference being that the gradient $\nabla f(.)$ gets replaced by the *gradient mapping* $g_{\mathbb{C}}(.)$ of f on set $\mathbb{C}$,

given by

$$g_{\mathbb{C}}(\bar{z}) = L(\bar{z} - z_{\mathbb{C}}(\bar{z})) \tag{10a}$$

$$z_{\mathbb{C}}(\bar{z}) = \arg\min_{z\in\mathbb{C}} \left\| z - z_f(\bar{z}) \right\|^2 \tag{10b}$$

$$z_f(\bar{z}) = \bar{z} - \frac{1}{L}\nabla f(\bar{z}) \ , \tag{10c}$$

for some $\bar{z} \in \mathbb{R}^p$. The vector $z_{\mathbb{C}}(\bar{z})$ can be interpreted as the projection of a gradient step taken at $\bar{z}$ with step size $1/L$ onto the feasible set $\mathbb{C}$. This operation is central to computing the gradient mapping and defines a *simple set* as a convex set for which one can compute this projection easily. Such simple sets are for instance the n-dimensional Euclidean ball, positive orthant and box.

B. Fast Gradient Methods

Fast, or optimal gradient methods were developed by Yurii Nesterov in 1983 [8]. The basic idea underlying these methods is to drop the stringent condition of forming a relaxation sequence (9). Instead, fast gradient methods make use of so-called estimate sequences:

Definition 3 (Estimate Sequence [11]): A pair of sequences $\{\phi_i(z)\}_{i=0}^{\infty}$ and $\{\lambda_i\}_{i=0}^{\infty}$, $\lambda_i \geq 0$ is called an estimate sequence of function f if $\lambda_i \to 0$ and

$$\phi_i(z) \leq (1-\lambda_i) f(z) + \lambda_i \phi_0(z) \ , \quad \forall i \geq 0, \ \forall z \in \mathbb{C} \ .$$

The next lemma demonstrates why it is useful for a gradient method to rely on a relaxation sequence.

Lemma 1: (cf. [11, §2.2.1]) Let the pair of sequences $\{\phi_i(z)\}_{i=0}^{\infty}$, $\{\lambda_i\}_{i=0}^{\infty}$ form an estimate sequence of function f. If for some sequence $\{z_i\}$, $z_i \in \mathbb{C}$ it holds

$$f(z_i) \leq \phi_i^* \equiv \min_{z\in\mathbb{C}} \phi_i(z) \ ,$$

then $f(z_i) - f^* \leq \lambda_i [\phi_0(z^*) - f^*]$.

Lemma 1 states that the rate of convergence of sequence $\{f(z_i) - f^*\}$ is identical to the rate of convergence of sequence $\{\lambda_i\}$ and that the initial residual is proportional to the difference $\phi_0(z^*) - f^*$; z^* being the optimal solution to (5), i.e. $f^* = f(z^*)$.

For a fast gradient method to work, one has to define an estimate sequence of function f as well as ensure the premise in Lemma 1. The former can be done by defining function ϕ_0 as the quadratic

$$\phi_0(z) = \phi_0^* + \frac{\gamma_0}{2} \|z - v_0\|^2 \ , \quad \gamma_0 > 0 \ , \tag{11}$$

and updating only the parameters ϕ_i^*, γ_i and v_i for all $i \geq 1$ following the recursive update rules given in [11, §2.2.1]. These rules require the (mapped) gradient as is shown in [11, §2.2.1], which validates the name 'gradient method' for the resulting scheme.

We are now ready to state the general iterative scheme of a fast gradient method with constant step size following [11, §2.2.3]. We consider the more general case of constrained minimization since it includes unconstrained minimization as a special case:

Algorithm 1 Fast Gradient Method for Constrained Minimization

Require: Initial point $z_0 \in \mathbb{C}$, $y_0 = z_0$, $0 < \sqrt{\frac{\mu}{L}} \leq \alpha_0 < 1$, Lipschitz constant L, convexity parameter μ, $i = 0$

1: **loop**
2: $z_{i+1} = z_{\mathbb{C}}(y_i)$, using (10b)
3: Compute $\alpha_{i+1} \in (0,1)$ from $\alpha_{i+1}^2 = (1-\alpha_{i+1})\alpha_i^2 + \mu\alpha_{i+1}/L$
4: $\beta_i = (\alpha_i(1-\alpha_i)) / (\alpha_i^2 + \alpha_{i+1})$
5: $y_{i+1} = z_{i+1} + \beta_i(z_{i+1} - z_i)$
6: $i = i + 1$
7: **end loop**

Remark 2: Most notably, the complexity of each iteration of the fast gradient scheme is of the same order as the complexity of traditional gradient methods.

C. Convergence Results

For convex optimization problems of type (5) with a strongly convex objective function f with Lipschitz continuous gradient, gradient methods allow for a global convergence analysis for both the unconstrained and the constrained case. Not only is it possible to guarantee convergence but also to specify the rate of convergence.

Definition 4 (Linear Convergence): A sequence $\{r_i \geq 0\}$ converges linearly to 0 with convergence ratio $q \in (0,1)$, if for some constant $K < \infty$

$$r_i \leq q^i K, \quad \forall i = 1, 2 \ldots \ .$$

Definition 5 (Sublinear Convergence): A sequence $\{r_i \geq 0\}$ that converges to 0, but is not linearly converging, is called sublinearly converging.

In the context of gradient methods it is natural to define the entities of the sequence $\{r_i\}$, let us call them *residuals*, as

$$r_i = f(z_i) - f^* \ \geq 0 \, ,$$

where z_i is the ith iterate of the gradient method.

Traditional gradient methods show global linear convergence of this sequence of residuals with a convergence ratio $1 - O(1/Q_f)$, where the condition number is defined as $Q_f := L/\mu$, $Q_f \geq 1$. Usually, a termination criterion $r_i \leq \varepsilon$, $\varepsilon > 0$ is used. In order to meet this criterion it follows from the convergence analysis that $O(1)Q_f \ln(1/\varepsilon)$ iterations are required [11, §2]. In practice, the condition number Q_f can be several orders of magnitude in size which may lead to a prohibitively large number of iterations.

Fast gradient methods also converge linearly and globally but with a much smaller convergence ratio $1 - O(1/\sqrt{Q_f})$, which allows one to reach an ε-solution in $O(1)\sqrt{Q_f}\ln(1/\varepsilon)$ iterations. Specifically, for Algorithm 1 the following convergence result holds:

Theorem 1 (Convergence of Algorithm 1 [11]): The sequence of iterates $\{z_i\}$ obtained from Algorithm 1 generates a sequence of residuals $\{r_i\}$ whose elements satisfy

$$r_i \leq \min\left\{ \left(1 - \sqrt{\frac{\mu}{L}}\right)^i, \frac{4L}{\left(2\sqrt{L} + i\sqrt{\gamma_0}\right)^2} \right\} [\phi_0(z^*) - f^*] \ , \tag{12}$$

for all $i \geq 0$, where

$$\gamma_0 = \frac{\alpha_0(\alpha_0 L - \mu)}{1 - \alpha_0} \ . \tag{13}$$

Remark 3: The condition $\alpha_0 \geq \sqrt{\mu/L}$ in Algorithm 1 corresponds to condition $\gamma_0 \geq \mu$, where γ_0 is the initial parameter in (11). This condition is necessary to obtain the convergence result of Theorem 1 as shown in [11, §2.2.1].

Remark 4: According to Theorem 1 the rate of convergence of residuals $\{r_i\}$ is the best of either a linear or a sublinear convergence rate.

The convergence result for Algorithm 1 given by Theorem 1 can be shown to be optimal in the sense that relying only on gradient information, one cannot find better convergence ratios for the class of optimization problems considered here. Details can be found in [11, §2.1].

IV. Fast Gradient Methods in MPC

We are now ready to apply the fast gradient scheme of Section III-B to obtain an ε-solution U_ε to the MPC problem for a given state x, i.e. a solution that satisfies

$$J_N(U_\varepsilon; x) - J_N^*(x) \leq \varepsilon \ , \quad \varepsilon > 0 \ , \tag{14}$$

where $J_N^*(x)$ is the value of the optimal solution.

For this problem we investigate two different strategies that differ in the choice of the initial iterate z_0 in Algorithm 1. Let us adapt notation used in control from now on, for instance denote the initial iterate z_0 in the MPC context by U_0.

The first strategy, that we refer to as *cold-starting*, is based on picking a fixed, state-independent sequence of control inputs $U_c \in \mathbb{U}^N$ which will allow us to obtain an admissible initial iterate U_0 – in the sense of Lemma 1 – with a single projection operation. In Section IV-B it will be shown that this strategy considerably simplifies the complexity analysis of the resulting fast gradient scheme.

The obvious alternative strategy is to re-start the algorithm from an ε-solution that was obtained at the previous time-step. This strategy is called *warm-starting* and different strategies are possible to make use of the previous solution. One of these possibilities will be examined in Section IV-C showing that the complexity of solving MPC problem (3) is linked to entities that are common in optimal control.

For both cold- and warm-starting strategies we derive upper bounds on the number of iterations required to obtain an ε-solution for any state $x \in \mathbb{X}$, where set $\mathbb{X} \subset \mathbb{R}^n$ is a compact, convex set of states that is defined for the specific application.

Before we go to the analysis, we will point out that the main ingredients of Algorithm 1, i.e. projection on a convex set and *a priori* computation of Lipschitz constant L and convexity parameter μ can be obtained very easily in the context of the considered MPC problem as will be explained in the following sections.

A. Computational Aspects of Fast Gradient Method in MPC

1) Projection on Convex Set: Recalling the fast gradient scheme that is given by Algorithm 1 we observe that in each iteration we need to compute the projection in (10b). The problem translates to

$$U_{\mathbb{U}^N}(\bar{U}) = \arg\min_{U \in \mathbb{U}^N} \left\| U - U_f(\bar{U}) \right\|^2 \tag{15a}$$

$$U_f(\bar{U}) = \bar{U} - \frac{1}{L} \nabla J_N(\bar{U}; x) \ , \tag{15b}$$

in the context of MPC problem (3). We observe that the feasible set in this problem is the direct product of N lower-dimensional sets, $\mathbb{U}^N = \mathbb{U} \times \mathbb{U} \ldots \times \mathbb{U}$. Since the objective in (15a) can be separated accordingly into N independent objectives, it suffices to consider minimization problems

$$u_{\mathbb{U},k} = \arg\min_{u \in \mathbb{U}} \left\| u - u_{f,k} \right\|^2, \quad k = 0 \ldots N-1 \ , \tag{16}$$

only, where the original solution $U_{\mathbb{U}^N}(\bar{U})$ is obtained by stacking all N solutions to (16), i.e. $U_{\mathbb{U}^N}(\bar{U}) = \left(u_{\mathbb{U},0}, u_{\mathbb{U},1}, \ldots, u_{\mathbb{U},N-1}\right)$. The vectors $u_{f,k}$ in (16) are obtained by truncating $U_f(\bar{U})$ into N m-dimensional vectors $u_{f,k}$, i.e. $\left(u_{f,0}, u_{f,1}, \ldots, u_{f,N-1}\right) = U_f(\bar{U})$.

The only challenge left is to solve the set of minimization problems (16) efficiently. Fortunately, in control, we often quite naturally encounter upper/lower bounds on the control inputs that lead to an m-dimensional box $\mathbb{U}$. Box constraints allow one to compute the projection (16) analytically by saturating every component j of $u_{f,k}$ according to the j*th* upper/lower bound of box $\mathbb{U}$.

Remark 5: Even in the case when the set of feasible inputs $\mathbb{U}$ is an arbitrary polytope, there is the possibility of pre-computing an explicit solution to (16) with $u_{f,k}$ being the parameter by means of multi-parametric programming [12].

2) Computation of Lipschitz Constant and Convexity Parameter: In general, it suffices to have an upper bound on the Lipschitz constant L and a lower bound on the convexity parameter μ in order to make the algorithmic scheme of fast gradient methods work. Of course, the convergence rates are best if the bounds for both parameters are tight. We will see that tight bounds can be achieved in the case of MPC problem (3). In order to show this, we make use of the fact that function $J_N(U;x)$ is twice continuously differentiable in U, so that we can apply the following lemmas.

Lemma 2 (cf. [11]): Let function f be twice continuously differentiable on set $\mathbb{C}$. The gradient ∇f is Lipschitz continuous on set $\mathbb{C}$ with Lipschitz constant L if and only if

$$\left\| \nabla^2 f(z) \right\| \leq L \ , \quad \forall z \in \mathbb{C} \ .$$

Matrix $\nabla^2 f(.)$ denotes the symmetric $p \times p$ Hessian matrix of function f and $\|.\|$ denotes the induced Euclidean norm, also called maximum singular value.

Lemma 3 (cf. [11]): Let function f be twice continuously differentiable on set $\mathbb{C}$. Function f is strongly convex on set $\mathbb{C}$ with convexity parameter μ if and only if there exists $\mu > 0$ such that

$$\nabla^2 f(z) \geq \mu I \ , \quad \forall z \in \mathbb{C} \ .$$

Lemmas 2 and 3 indicate to us how to compute Lipschitz constant L and convexity parameter μ, as shown next.

Proposition 1: For the MPC problem in (3), the Lipschitz constant L of the gradient $\nabla J_N(U;x)$ and the convexity parameter μ of the objective function $J_N(U;x)$ are independent of the state x and are given by

$$L = \lambda_{\max}(\mathscr{T}) \ , \qquad \mu = \lambda_{\min}(\mathscr{T}) \ ,$$

where $\lambda_{\max}$ ($\lambda_{\min}$) denotes the maximum (minimum) eigenvalue of the Hessian matrix $\mathscr{T}$ of function $J_N(U;x)$ in (4).

Proof: Hessian $\mathscr{T}$ is independent of the variable U and state x in MPC problem (3). Since $\mathscr{T}$ is symmetric and positive definite under $Q \geq 0, \ R > 0$, the maximum singular value coincides with the maximum eigenvalue $\lambda_{\max}(\mathscr{T})$. For the same reasons $\mathscr{T} \geq \lambda_{\min}(\mathscr{T})I$ with $\lambda_{\min}(\mathscr{T}) > 0$. ■

3) Computational Complexity per Iteration: The main complexity of an iteration of Algorithm 1 stems from computing the gradient $\nabla J_N(U;x) = \mathscr{T}U + \mathscr{L}^T x$ in (15b), since Hessian $\mathscr{T}$ is a dense matrix in general. The matrix-vector product $\mathscr{T}U$ needs approximately $2(Nm)^2$ floating point operations (*flops*). Assuming that vector $\mathscr{L}^T x$ is computed only once per time-step, computation of the gradient $\nabla J_N(U;x)$ requires $2(Nm)^2 + Nm$ flops in each iteration. The total effort required for every iteration of the fast gradient scheme given by Algorithm 1 when applied to the MPC problem (3) is $2(Nm)^2 + 7Nm$ flops per iteration assuming box constraints and neglecting the computation of scalar variables.

An interior point approach that makes use of the special structure of the MPC problem as discussed by [6] has a complexity of $O\left(N(n+m)^3\right)$ flops per iteration and in the case of diagonal weighting matrices Q, R and box constraints improves to $O\left(N(n^3 + n^2 m)\right)$. In the context of these methods an iteration consists of computing a Newton step and practice shows that the total number of iterations required is in the order of tens, unless appropriate measures are taken to decrease this number [6]. However, no theoretical evidence is given at the moment that underlines these practical results. The same holds true for active set methods, which are exponential in the worst case but also show good practical performance. If efficiently implemented, active set methods allow one to compute an iteration in $O\left((Nm)^2\right)$ flops, where the main work is spent in solving the KKT system [3], [4].

B. Coldstart Fast Gradient Method

We are now ready to derive upper bounds on the total number of iterations required to obtain an ε-solution U_ε defined by (14) for the MPC problem (3). We will perform the analysis under the assumption that at every time-step the same fixed sequence of controls $U_c \in \mathbb{U}^N$ is provided, a strategy that we call cold-starting.

In view of Algorithm 1 and its convergence result in Theorem 1 as well as the results from Section IV-A we realize, that the remaining unknown entities for bounding the total number of iterations by means of (12) are the initial residual $\phi_0(U^*;x) - J_N^*(x)$ and the value of γ_0. The sequence of controls U^* denotes the optimal input sequence for problem (3) with initial state x satisfying $J_N^*(x) = J_N(U^*;x)$.

The following analysis will provide a way to upper-bound this initial residual by making use of compactness of the feasible set of inputs $\mathbb{U}$. We start the analysis with the definition of an appropriate quadratic function ϕ_0[1]:

Lemma 4: Let $U_c \in \mathbb{U}^N$ and L be the Lipschitz constant of function $J_N(U;x)$ given by (4). The choice $\phi_0 = \hat{\phi}_0$, where

$$\hat{\phi}_0(U;x) := J_N(U_c;x) + \nabla J_N(U_c;x)^T (U - U_c) + \frac{L}{2}\|U - U_c\|^2$$

provides an upper bound of the initial residual in (12) by

$$\hat{\phi}_0(U^*;x) - J_N^*(x) \leq \frac{L}{2}\|U^* - U_c\|^2 . \tag{17}$$

Proof: We evaluate function $\hat{\phi}_0(U;x)$ at point $U = U^*$ and apply relation (6) to obtain (17). ■

Note that U_c must not be chosen as the initial iterate in Algorithm 1 since the initial iterate has to satisfy Lemma 1. We show next how to obtain an admissible initial iterate U_0.

Lemma 5: Let $\hat{\phi}_0$ be defined as in Lemma 4 with $U_c \in \mathbb{U}^N$. An admissible initial iterate U_0 in the sense of Lemma 1 is given by

$$U_0 = U_{\mathbb{U}^N}(U_c) \ ,$$

where $U_{\mathbb{U}^N}(.)$ is defined by (15a).

Proof: According to Lemma 1 we have to show that

$$J_N(U_0;x) \leq \hat{\phi}_0^* \equiv \min_{U \in \mathbb{U}^N} \hat{\phi}_0(U;x) . \tag{18}$$

By considering (7) we observe that $\hat{\phi}_0$ upper-bounds $J_N(U;x)$

$$J_N(U;x) \leq \hat{\phi}_0(U;x), \quad \forall U \in \mathbb{U}^N \ ,$$

such that by choosing

$$U_0 = \arg\min_{U \in \mathbb{U}^N} \hat{\phi}_0(U;x) \ , \tag{19}$$

we certainly fulfill (18). It is left to show that (19) can be rewritten as the projection given by (15a). For this we simply observe that $\hat{\phi}_0$ can be reformulated as

$$\hat{\phi}_0(U;x) = \hat{\phi}_{f,0}^* + \frac{L}{2}\left\|U - U_f(U_c)\right\|^2 \ ,$$

by developing the Taylor series of $\hat{\phi}_0$ at the point $U_f(U_c)$ where the unconstrained minimum $\hat{\phi}_{f,0}^*$ is attained. ■

We are now ready to provide an upper bound on the number of iterations in the case of a cold-starting strategy:

Proposition 2: Let $U_c \in \mathbb{U}^N$ be a fixed sequence of control inputs and the initial iterate U_0 in Algorithm 1 be chosen according to Lemma 5. For the MPC problem in (3) an ε-solution U_ε defined by (14) can be obtained after at most

$$I_{\max} = \min\left\{ \left\lceil \frac{\ln 2\varepsilon - \ln Ld^2}{\ln\left(1 - \sqrt{\frac{\mu}{L}}\right)} \right\rceil , \left\lceil \sqrt{\frac{2Ld^2}{\varepsilon}} - 2 \right\rceil \right\}$$

iterations where d^2 is given by

$$d^2 = \max_{U \in \mathbb{U}^N} \|U - U_c\|^2 \ . \tag{20}$$

[1]Private communication with Y. Nesterov.

Proof: Going back to Theorem 1 we observe that a sufficient condition for an ε-solution U_ε is

$$\min\left\{\left(1-\sqrt{\frac{\mu}{L}}\right)^i, \frac{4L}{\left(2\sqrt{L}+i\sqrt{\gamma_0}\right)^2}\right\}\left[\hat{\phi}_0(U^*;.)-J_N^*(.)\right]\leq\varepsilon$$

Defining $\hat{\phi}_0$ as in Lemma 4 we conclude that $\gamma_0 = L$ and that the initial residual can be upper-bounded by $L\|U^*-U_c\|^2/2$ which is upper-bounded by $Ld^2/2$ under the assumption that the feasible set of inputs $\mathbb{U}$ is compact. Putting all together gives the upper bound $I_{\max}$ on the number of iterations. ■

Remark 6: If the fixed input sequence $U_c \in \mathbb{U}^N$ is chosen as the center of set $\mathbb{U}^N$, the maximum d^2 of problem (20) refers to the squared radius of set $\mathbb{U}^N$ and thus is easily computed in general.

C. Warmstart Fast Gradient Method

An alternative strategy to cold-starting as discussed in Section IV-B is to find an initial iterate for the fast gradient scheme based on the solution from the previous time-step. This strategy is referred to as warm-starting and is generally used by optimization methods if a *reasonable guess* about the solution can be made beforehand, hoping that the effort needed to solve the problem becomes less than starting from a fixed point. Warm-starting is also used in the interior point approach by [6] and the active set method by [4], but it is generally unclear how to quantify the benefit obtained by warm-starting *a priori* for these methods. We will show that for fast gradient methods it is possible to specify this benefit and also to interpret the result in the language of optimal control.

Intuitively, in the context of MPC, we expect warm-starting to be a promising approach for the following reason: Having worked out a close to optimal policy for a given state over N steps into the future and having applied the first one, the remaining $N-1$-step policy should be close to optimality for the successor state. Of course, we ask for an optimal N-step policy; Nevertheless, re-using the previous $N-1$-step plan might still be useful if one appends a meaningful *Nth* element to it. One can think of several possibilities for this whereby for the upcoming analysis a generic warm-starting strategy is defined as follows:

Definition 6: Let the sequence of control inputs $(u_0, u_1, \ldots, u_{N-1}) \in \mathbb{U}^N$ be a feasible ε-solution in the sense of (14) to the MPC problem (3) with initial state x^-. Assume, that the first input of this sequence u_0 is applied to the plant and results in state x at the next time-step. Then

$$U_w := (u_1, \ldots, u_{N-1}, u_N) \quad , \quad u_N \in \mathbb{U} \ ,$$

defines a feasible warm-starting sequence of inputs for the MPC problem (3) with initial state x.

Now, the same algorithmic framework as discussed in Section IV-B with $U_c = U_w$ is applicable. In fact, Proposition 2 will give the desired maximum number of iterations once a problem similar to (20) is solved. Unfortunately, apart from being a hard problem no insight with respect to the optimal control terminology can be gained from its solution. Therefore we introduce a slightly different approach next, that will allow us to get an intuition on what we can expect in terms of number of iterations to solve MPC problem (3) by applying a certain warm-starting technique.

Before doing so, we define a certificate for optimality in constrained optimization (cf. e.g. [11, §2.2.2]):

Theorem 2: Let f be once continuously differentiable and $\mathbb{C}$ a closed convex set. Point $z^* \in \mathbb{C}$ is an optimal solution to (5) if and only if

$$\nabla f(z^*)^T(z-z^*) \geq 0 \ , \quad \forall z \in \mathbb{C} \ .$$

Lemma 6: Let $U_w \in \mathbb{U}^N$ be a feasible sequence of inputs and μ be the convexity parameter of function $J_N(U;x)$ given by (4). The choice

$$\hat{\phi}_0(U;x) := J_N(U_w;x) + \frac{\mu}{2}\|U-U_w\|^2$$

admits to upper-bound the initial residual in (12) by

$$\hat{\phi}_0(U^*;x) - J_N^*(x) \leq 2\left(J_N(U_w;x) - J_N^*(x)\right) \ . \qquad (21)$$

Also, input sequence U_w is an admissible initial iterate in the sense of Lemma 1 for Algorithm 1, i.e. $U_0 = U_w$.

Proof: For the upper bound on the initial residual we exploit strong convexity of the function $J_N(U;x)$, so that

$$\begin{aligned} J_N(U_w;x) &\geq J_N^*(x) + \nabla J_N(U^*;x)^T(U_w-U^*) + \frac{\mu}{2}\|U_w-U^*\|^2 \\ &\geq J_N^*(x) + \frac{\mu}{2}\|U_w-U^*\|^2 \ , \end{aligned}$$

follows from (8) and Theorem 2. Using this result in

$$\hat{\phi}_0(U^*;x) - J_N^*(x) = J_N(U_w;x) - J_N^*(x) + \frac{\mu}{2}\|U_w-U^*\|^2 \ ,$$

gives the upper bound in Lemma 6. The input sequence U_w is an admissible initial iterate since by definition of $\hat{\phi}_0$

$$U_w = \arg\min_{U\in\mathbb{U}^N} \hat{\phi}_0(U;x),$$

which satisfies the premise of Lemma 1. ■

Lemma 6 allows one to obtain an upper bound on the number of iterations in the case of a warm-starting strategy:

Proposition 3: Let $U_w \in \mathbb{U}^N$ be given by Definition 6 and $\mathbb{X} \subset \mathbb{R}^n$ be the set of initial states x. For the MPC problem in (3) an ε-solution U_ε defined by (14) can be obtained after at most

$$I_{\max} = \min\left\{\left\lceil\frac{\ln\varepsilon - \ln 2\delta}{\ln\left(1-\sqrt{\frac{\mu}{L}}\right)}\right\rceil, \left\lceil\frac{1}{\sqrt{\mu}}\left(\sqrt{\frac{8\delta L}{\varepsilon}}-2\sqrt{L}\right)\right\rceil\right\}$$

iterations if the initial iterate is chosen as $U_0 = U_w$ in Algorithm 1. Constant δ is defined as

$$\begin{aligned} \delta = \max \ & J_N(U_w;x) - J_N^*(x) \qquad (22) \\ \text{subject to } & U_w \in \mathbb{U}^N \text{ (according to Definition 6)} \\ & x \in \mathbb{X} \ . \end{aligned}$$

Proof: The proof follows closely the proof of Proposition 2 whereby here Lemma 6 is used to upper-bound the initial residual (12) and also fixes $\gamma_0 = \mu$. Finally, the maximization problem (22) bounds the worst case residual over all initial states $x \in \mathbb{X}$. ■

Remark 7: Due to lack of space maximization problem (22) is only defined informally. This problem is a non-convex multi-level optimization problem that contains optimizers of other optimization problems as decision variables. It can easily be defined such that a mismatch between the real and the predicted successor state can be considered. Also, one can take into account that the warm-starting sequence U_w stems from an ε-solution from the previous time-step. Various solution methods for problem (22) exist in the literature, see e.g. [13] for an overview. In [14] a way to rewrite problems of type (22) as a mixed-integer linear program is discussed and was the author's choice for solving this problem.

The careful reader might have noticed that the discussion so far has not revealed an intuitive interpretation on computational complexity in the language of optimal control yet, as promised in the introduction of this section. Looking at Proposition 3 the only entity that prevents one from developing intuition, is the constant δ given by maximization problem (22). Interestingly, it turns out that in case of a specific variant of warm-starting and under additional assumptions this value has the following interpretation.

Proposition 4: Assume that the optimal solution to the MPC problem (3) at the previous time-step is known and that the predicted successor state is equivalent to the real successor state $x \in \mathbb{X}$. Given a warm-starting sequence U_w defined by Definition 6 where we choose $u_N = 0$ we obtain $J_N(U_w;x) - J_N^*(x) = J_{N-1}^*(x) - J_N^*(x)$.

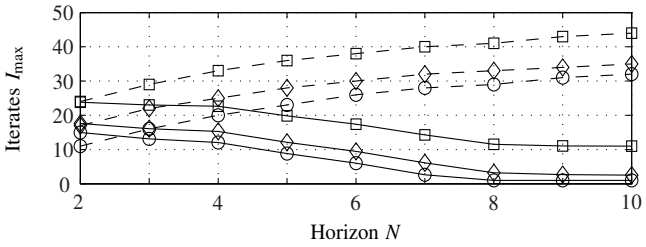
Proof: From optimality of the previous solution over a horizon of length N and equivalence of predicted and real successor state x we infer from the principle of optimality that the remaining sequence of $N-1$ optimal inputs is optimal for the MPC problem with horizon $N-1$ and initial state x. This gives cost $J_{N-1}^*(x)$ that is equal to cost $J_N(U_w;x)$ with U_w chosen as stated in Proposition 4. ■

Proposition 4 now means that one can expect – admittedly in a very idealistic setting – the difference $J_{N-1}^*(x) - J_N^*(x)$ to approach 0 if the control horizon N is chosen *large enough*. On the one hand, this result is surprising, on the other hand it is not since for the limit case, i.e. $N \to \infty$ the "N"-step policy coincides with the "$N-1$"-step policy.

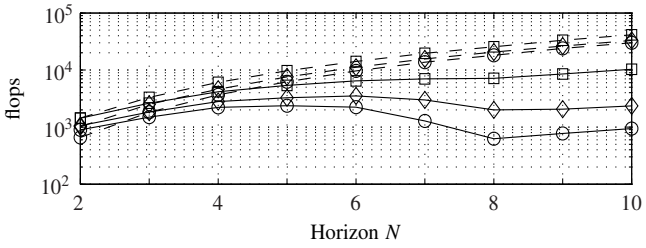
V. Illustrative Example

We consider the four-state/two-input system given in [2], restricting the initial state to $\|x\|_\infty \leq 10$ and input to $\|u\|_\infty \leq 1$. The weighting matrices are chosen as $Q = I$ and $R = 0.1I$.

Figs. 1(a) and 1(b) show two families of plots, parameterized by ε, that depict the dependence of the maximum number of iterations and maximum floating point operations respectively, needed for cold-starting with $U_c = 0$ and warm-starting with $u_N = 0$, on the horizon N. For warm-starting we assume the maximum mismatch $\Delta x \in \mathbb{R}^n$ between real and predicted successor state to be bounded by $\|\Delta x\|_\infty \leq 0.25$ (see Remark 7). From the plots we observe that the cold-starting strategy requires monotonically increasing computational efforts to obtain a solution whereas warm-starting does not show this behavior as expected from Proposition 4.



(a) Maximum number of iterations $I_{\max}$.



(b) Maximum number of flops.

Fig. 1. Computational complexity of cold-starting (*dashed*) and warm-starting (*solid*) for different horizon lengths in Example V, parameterized by $\varepsilon = 0.1$ (*circle*), $\varepsilon = 0.05$ (*diamond*) and $\varepsilon = 0.01$ (*square*).

VI. Acknowledgements

The authors would like to thank Yurii Nesterov for many valuable discussions during the preparation of this paper.

References

[1] A. Bemporad, M. Morari, V. Dua, and E. Pistikopoulos, "The Explicit Linear Quadratic Regulator for Constrained Systems," *Automatica*, vol. 38, no. 1, pp. 3–20, Jan. 2002.

[2] C. Jones and M. Morari, "The Double Description Method for the Approximation of Explicit MPC Control Laws," in *Conference on Decision and Control, CDC*, Cancun, Mexico, Dec. 2008.

[3] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York: Springer, 2006.

[4] H. J. Ferreau, H. G. Bock, and M. Diehl, "An online active set strategy to overcome the limitations of explicit MPC," *International Journal of Robust and Nonlinear Control*, vol. 18, pp. 816–830, 2008.

[5] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, Mar. 2004.

[6] Y. Wang and S. Boyd, "Fast Model Predictive Control Using Online Optimization," in *Proceedings of the 17th World Congress*. Seoul: IFAC, 2008.

[7] M. Zeilinger, C. Jones, and M. Morari, "Real-time suboptimal Model Predictive Control using a combination of Explicit MPC and Online Computation," in *Conference on Decision and Control, CDC*, Cancun, Mexico, Dec. 2008.

[8] Y. Nesterov, "A method for solving a convex programming problem with convergence rate $1/k^2$," *Soviet Math. Dokl.*, vol. 27, no. 2, pp. 372–376, 1983.

[9] A. Glattfelder and W. Schaufelberger, *Control Systems with Input and Output Constraints*. London: Springer, 2003.

[10] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, June 2000.

[11] Y. Nesterov, *Introductory Lectures on Convex Optimization : A Basic Course*, Kluwer Acad. Publ., 2004.

[12] B. Bank, J. Guddat, D. Klatte, B. Kummer, and K. Tammer, *Non-Linear Parametric Optimization*. Berlin: Akademie-Verlag, 1982.

[13] B. Colson, P. Marcotte, and G. Savard, "Bilevel programming: A survey," *4OR: A Quarterly Journal of Operations Research*, vol. 3, no. 2, pp. 87–107, June 2005.

[14] C. Jones and M. Morari, "Approximate Explicit MPC using Bilevel Optimization," in *European Control Conference*, Budapest, Hungary, Aug. 2009.