Delft University of Technology

Delft Center for Systems and Control

Technical report 12-011

A distributed optimization algorithm with convergence rate $O(\frac{1}{k^2})$ for distributed model predictive control*

P. Giselsson, M.D. Doan, T. Keviczky, B. De Schutter, and A. Rantzer

March 2012

Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 CD Delft
The Netherlands
phone: +31-15-278.24.73 (secretary)
URL: https://www.dcsc.tudelft.nl

* This report can also be downloaded via https://pub.bartdeschutter.org/abs/12_011.html

A distributed optimization algorithm with convergence rate $O(\frac{1}{k^2})$ for distributed model predictive control

Pontus Giselsson* **Minh Dang Doan**** **Tamás Keviczky****
Bart De Schutter** **Anders Rantzer***

* *Department of Automatic Control, Lund University, Sweden*
(email: {pontusg,rantzer}@control.lth.se)
** *Delft Center for Systems and Control, Delft University of Technology, The Netherlands*
(email: {m.d.doan,t.keviczky,b.deschutter}@tudelft.nl)

Abstract

We propose a distributed optimization algorithm for mixed $\mathcal{L}_1/\mathcal{L}_2$-norm optimization based on accelerated gradient methods using dual decomposition. The algorithm achieves convergence rate $O(\frac{1}{k^2})$, where k is the iteration number, which significantly improves the convergence rates of existing duality-based distributed optimization algorithms that achieve $O(\frac{1}{k})$. The performance of the developed algorithm is evaluated on randomly generated optimization problems arising in distributed Model Predictive Control (MPC). The evaluation shows that, when the problem data is sparse and large-scale, our algorithm outperforms state-of-the-art optimization software CPLEX and MOSEK.

Keywords: Distributed optimization, Accelerated gradient algorithm, Model predictive control

1. Introduction

Gradient-based optimization methods are known for their simplicity and low complexity within each iteration. A limitation of classical gradient-based methods is the slow rate of convergence. It can be shown [1, 2] that for functions with a Lipschitz-continuous gradient, i.e. smooth functions, classical gradient-based methods converge at a rate of $O(\frac{1}{k})$, where k is the iteration number. In [3] it was shown that a lower bound on the convergence rate for gradient-based methods is $O(\frac{1}{k^2})$. Nesterov showed in his work [4] that an accelerated gradient algorithm can be constructed such that this lower bound on the convergence rate is achieved when minimizing unconstrained smooth functions. This result has been extended and generalized in several publications to handle constrained smooth problems and smooth problems with an additional non-smooth term [5, 6, 7, 8]. Gradient-based methods are suitable for distributed optimization when they are used in combination with dual decomposition techniques.

Dual decomposition is a well-established concept since around 1960 when Uzawa's algorithm [9] was presented. Similar ideas were exploited in large-scale optimization [10]. Over the next decades, methods for decomposition and coordination of dynamic systems were developed and refined [11, 12, 13] and used in large-scale applications

[14]. In [15] a distributed asynchronous method was studied. More recently dual decomposition has been applied in the distributed Model Predictive Control (MPC) literature in [16, 17, 18, 19] for problems with a strongly convex quadratic cost and arbitrary linear constraints. The above mentioned methods rely on gradient-based optimization, which suffers from slow convergence properties $O(\frac{1}{k})$. Also the step size parameter in the gradient scheme must be chosen appropriately to get good performance. Such information has not been provided or has been chosen conservatively in these publications.

In this work we improve on the previously presented distributed optimization methods by using the accelerated gradient method to solve the dual problem instead of the classical gradient method. We also extend the class of problems considered by allowing an additional sparse but non-separable 1-norm penalty. Such 1-norm terms are used as regularization term or as penalty for soft constraints [20]. Further, we also provide the optimal step-size parameter for the algorithm, which is crucial for performance. The convergence rate for the dual function value using the accelerated gradient method is implicitly known from [7, 8]. This convergence rate in the dual function value does, however, not indicate the rate at which the primal iterate approaches the primal optimal solution. In this paper we also provide convergence rate results for the primal variables.

Related to our work is the approach presented in [21] for systems with a (non-strongly) convex cost. It is based on the smoothing technique presented by Nesterov in [6]. Other relevant work is presented in [22, 23] in which optimization problems arising in MPC are solved in a centralized fashion using accelerated gradient methods. These methods are, however, restricted to handle only box-constraints on the control signals.

To evaluate the proposed algorithm we solve randomly generated large-scale and sparse optimization problems arising in distributed MPC and compare the execution times to state-of-the-art optimization software for large-scale optimization, in particular CPLEX and MOSEK. We also evaluate the performance loss obtained when suboptimal step-lengths are used.

The paper is organized as follows. In Section 2, the problem setup is introduced. The dual problem to be solved is introduced in Section 3 and some properties of the dual function are presented. The distributed solution algorithm for the dual problem is presented in Section 4. In Section 5 a numerical example is provided, followed by conclusions drawn in Section 6.

2. Problem Setup

In this paper we present a distributed algorithm for optimization problems with cost functions of the form

$$J(x) = \frac{1}{2}x^T Hx + g^T x + \gamma\|Px - p\|_1. \tag{1}$$

The full decision vector, $x \in \mathbb{R}^n$, is composed of local decision vectors, $x_i \in \mathbb{R}^{n_i}$, according to $x = [x_1^T, \ldots, x_M^T]^T$. The quadratic cost matrix $H \in \mathbb{R}^{n\times n}$ is assumed separable, i.e. $H = \text{blkdiag}(H_1, \ldots, H_M)$ where $H_i \in \mathbb{R}^{n_i \times n_i}$. The linear part $g \in \mathbb{R}^n$ consists of local parts, $g = [g_1^T, \ldots, g_M^T]^T$ where $g_i \in \mathbb{R}^{n_i}$. Further $P \in \mathbb{R}^{m\times n}$ is composed of $P = [P_1, \ldots, P_m]^T$ where each $P_r \in \mathbb{R}^n$ which in turn consists of $P_r = [P_{r1}^T, \ldots, P_{rM}^T]^T$ with each $P_{ri} \in \mathbb{R}^{n_i}$. We do not assume that the matrix P should be block diagonal which means that the cost function J is not separable. However, we do assume that the vectors P_r have sparse structure. Sparsity refers to

the property that for each $r \in \{1, \dots, m\}$ there exist some $i \in \{1, \dots, M\}$ such that $P_{ri} = 0$. We also have $p = [p_1, \dots, p_m]^T$ and $\gamma > 0$. This gives the following equivalent formulation of (1)

$$J(x) = \sum_{i=1}^{M} \left[\frac{1}{2} x_i^T H_i x_i + g_i^T x_i \right] + \sum_{r=1}^{m} \left| \sum_{i=1}^{M} P_{ri}^T x_i - p_r \right|. \tag{2}$$

Minimization of (1) is subject to linear equality and inequality constraints

$$A_1 x = B_1 \qquad\qquad A_2 x \leq B_2$$

where $A_1 \in \mathbb{R}^{q \times n}$ and $A_2 \in \mathbb{R}^{(s-q) \times n}$ contain $a_l \in \mathbb{R}^n$ as $A_1 = [a_1, \dots, a_q]^T$ and $A_2 = [a_{q+1}, \dots, a_s]^T$. Further each $a_l = [a_{l1}^T, \dots, a_{lM}^T]^T$ where $a_{li} \in \mathbb{R}^{n_i}$. Further we have $B_1 \in \mathbb{R}^q$ and $B_2 \in \mathbb{R}^{s-q}$ where $B_1 = [b_1, \dots, b_q]^T$ and $B_2 = [b_{q+1}, \dots, b_s]^T$. We assume that the matrices A_1 and A_2 are sparse. By introducing the auxiliary variables x_a and the constraint $Px - p = x_a$ we get the following optimization problem:

$$\begin{aligned} \min_{x, x_a} \quad & \tfrac{1}{2} x^T H x + g^T x + \gamma \|x_a\|_1 \\ \text{s.t.} \quad & A_1 x = B_1 \\ & A_2 x \leq B_2 \\ & Px - p = x_a \end{aligned} \tag{3}$$

The objective of the optimization routine is to solve (3) in a distributed fashion using several computational units, where each computational unit computes the optimal local variables, x_i^*, only. Each computational unit is assigned a number of constraints in (3) that it is responsible for. We denote the set of equality constraints that unit i is responsible for by $\mathcal{L}_i^1$, the set of inequality constraints by $\mathcal{L}_i^2$ and the set of constraints originating from the 1-norm by $\mathcal{R}_i$. This division is obviously not unique but all constraints should be assigned to one computational unit. Further for $l \in \mathcal{L}_i^1$ and $l \in \mathcal{L}_i^2$ we require that $a_{li} \neq 0$ and for $r \in \mathcal{R}_i$ that $P_{ri} \neq 0$. Now we are ready to define two sets of neighbors to computational unit i:

$$\begin{aligned} \mathcal{N}_i = \big\{ j \in \{1, \dots, M\} \mid \; & \exists l \in \mathcal{L}_i^1 \text{ s.t. } a_{lj} \neq 0 \\ & \text{or } \exists l \in \mathcal{L}_i^2 \text{ s.t. } a_{lj} \neq 0 \\ & \text{or } \exists r \in \mathcal{R}_i \text{ s.t. } P_{rj} \neq 0 \big\} \\ \mathcal{M}_i = \big\{ j \in \{1, \dots, M\} \mid \; & \exists l \in \mathcal{L}_j^1 \text{ s.t. } a_{li} \neq 0 \\ & \text{or } \exists l \in \mathcal{L}_j^2 \text{ s.t. } a_{li} \neq 0 \\ & \text{or } \exists r \in \mathcal{R}_j \text{ s.t. } P_{ri} \neq 0 \big\} \end{aligned}$$

Through the introduction of these sets the constraints that are assigned to unit i can equivalently be written as

$$a_l^T x = b_l \Leftrightarrow \sum_{j \in \mathcal{N}_i} a_{lj}^T x_j = b_l, \qquad l \in \mathcal{L}_i^1 \tag{4}$$

$$a_l^T x \leq b_l \Leftrightarrow \sum_{j \in \mathcal{N}_i} a_{lj}^T x_j \leq b_l, \qquad l \in \mathcal{L}_i^2 \tag{5}$$

and the 1-norm term can equivalently be written as

$$|P_r^T x - p_r| = \Big| \sum_{j \in \mathcal{N}_i} P_{rj}^T x_j - p_r \Big|, \qquad r \in \mathcal{R}_i. \tag{6}$$

In the following section, the dual function to be maximized is introduced. First, we state some assumptions that will be useful in the continuation of the paper.

Assumption 1. *We assume that each H_i in* (2) *is a real symmetric positive definite matrix that satisfies the following eigenvalue bounds*

$$\underline{\sigma}_i I \preceq H_i \preceq \bar{\sigma}_i I$$

where $0 < \underline{\sigma}_i \leq \bar{\sigma}_i < \infty$.

Remark 1. *The corresponding bound for H becomes $\underline{\sigma} I \preceq H \preceq \bar{\sigma} I$ where $\underline{\sigma} := \min_i \underline{\sigma}_i$ and $\bar{\sigma} := \max_i \bar{\sigma}_i$. Also note that since H is positive definite we have $\frac{1}{\underline{\sigma}} I \preceq H^{-1} \preceq \frac{1}{\bar{\sigma}} I$ (c.f. [24, Corollary 7.7.4]).*

Assumption 2. *We assume that there exists a vector $\bar{x}$ such that $A_1\bar{x} = b_1$ and $A_2\bar{x} < b_2$. Further, we assume that $a_l, l = 1, \ldots, q$ and $P_r, r = 1, \ldots, m$ are linearly independent.*

Remark 2. *Assumption 2 is the Slater condition for* (3) *[1, Proposition 3.3.9] since we can always choose $\bar{x}_a = P\bar{x} - p$.*

3. Dual problem

In this section we introduce a dual problem to (3) from which the primal solution can be obtained. We show that this dual problem has the properties required to apply accelerated gradient methods. We also present some additional properties that are needed to prove convergence rates of the primal variables.

3.1. Formulation of the dual problem

We introduce Lagrange multipliers, $\lambda \in \mathbb{R}^q, \mu \in \mathbb{R}^{s-q}_{\geq 0}, v \in \mathbb{R}^m$ for the constraints in (3). Under Assumption 2 it is well known (cf. [25, §5.2.3]) that there is no duality gap and we get the following dual problem

$$\sup_{\lambda,\mu\geq 0,\nu} \inf_{x,x_a} \Big\{ \frac{1}{2} x^T H x + g^T x + \gamma \|x_a\|_1 + \lambda^T (A_1 x - B_1) + \\ + \mu^T (A_2 x - B_2) + \nu^T (Px - p - x_a) \Big\}. \tag{7}$$

After rearranging the terms and changing $\inf_x(\cdot)$ to $-\sup_x(-(\cdot))$ we get

$$\sup_{\lambda,\mu\geq 0,\nu} \Big\{ -\sup_x \Big[-(A_1^T\lambda + A_2^T\mu + P^T\nu + g)^T x - \frac{1}{2} x^T H x \Big] \\ - \lambda^T B_1 - \mu^T B_2 - \nu^T p - \sup_{x_a} \left[\nu^T x_a - \gamma \|x_a\|_1 \right] \Big\}. \tag{8}$$

The supremum over x_a can be solved explicitly:

$$\sup_{x_a} \left\{ \nu^T x_a - \gamma \|x_a\|_1 \right\} = \sup_{x_a} \left\{ \sum_i \left[\nu^i x_a^i - \gamma |x_a^i| \right] \right\}$$

$$= \sum_i \left\{ \sup_{x_a^i} \left[\nu^i x_a^i - \gamma |x_a^i| \right] \right\}$$
$$= \begin{cases} 0 & \text{if } \|\nu\|_\infty \leq \gamma \\ \infty & \text{else} \end{cases}$$

where super-script i denotes the i-th element in the vector. The supremum over x_a becomes a box-constraint for the dual variables ν. This is crucial for distribution reasons.

Before we explicitly solve the maximization over x in (8) the following notation is introduced

$$\mathcal{A} = [A_1^T \ A_2^T \ P^T]^T \qquad \mathcal{B} = [B_1^T \ B_2^T \ p^T]^T \qquad z = [\lambda^T \ \mu^T \ \nu^T]^T$$

where $\mathcal{A} \in \mathbb{R}^{(s+m)\times n}$, $\mathcal{B} \in \mathbb{R}^{s+m}$ and $z \in \mathbb{R}^{s+m}$. We also introduce the set of feasible dual variables

$$Z = \left\{ z \in \mathbb{R}^{s+m} \;\middle|\; \begin{array}{ll} z_l \in \mathbb{R} & l \in \{1, \ldots, q\} \\ z_l \geq 0 & l \in \{q+1, \ldots, s\} \\ |z_l| \leq \gamma & l \in \{s+1, \ldots, s+m\} \end{array} \right\} \tag{9}$$

Completion of squares in the maximization over x in (8) gives

$$\sup_x \left[-(\mathcal{A}^T z + g)^T x - \frac{1}{2} x^T H x \right] = \frac{1}{2} (\mathcal{A}^T z + g)^T H^{-1} (\mathcal{A}^T z + g)$$

and we get the following dual problem

$$\sup_{z \in Z} \left\{ -\frac{1}{2} (\mathcal{A}^T z + g)^T H^{-1} (\mathcal{A}^T z + g) - \mathcal{B}^T z \right\}. \tag{10}$$

We introduce the following definition of the negative dual function

$$f(z) := \frac{1}{2} (\mathcal{A}^T z + g)^T H^{-1} (\mathcal{A}^T z + g) + \mathcal{B}^T z.$$

It is easily seen that f is convex and differentiable with the following gradient

$$\nabla f(z) = \mathcal{A} H^{-1} (\mathcal{A}^T z + g) + \mathcal{B}. \tag{11}$$

Next we show some properties of the dual function. The proofs for Propositions 1 and 2 below can be found in the Appendix.

Proposition 1. *The gradient, ∇f, is Lipschitz continuous on Z with Lipschitz constant $L = \|\mathcal{A} H^{-1} \mathcal{A}^T\|_2$, i.e. for any $z_1 \in Z$ and $z_2 \in Z$ we have*

$$\|\nabla f(z_1) - \nabla f(z_2)\|_2 \leq L \|z_1 - z_2\|_2. \tag{12}$$

Further, the Lipschitz constant, L, is the smallest constant such that (12) *holds for all $z_1 \in Z$ and $z_2 \in Z$.*

Remark 3. *Note that it is known from [6, Theorem 1] that ∇f is Lipschitz continuous. However, the Lipschitz constant provided in [6], $\bar{L} = \frac{1}{\underline{\sigma}} \|\mathcal{A}\|^2$, is larger than the one presented here.*

Proposition 2. *Let Z^* denote the set of optimal dual variables, defined as*

$$Z^* = \{z^* \in Z \mid f(z^*) \leq f(z)\ \forall z \in Z\}$$

The following statements hold:

1. *Z^* is non-empty and bounded*
2. *For any $z^* \in Z^*$ and any $z \in Z$, we have*

$$f(z) - f(z^*) \geq \frac{1}{2\bar{\sigma}}\|\mathcal{A}^T(z - z^*)\|_2^2. \tag{13}$$

4. Distributed optimization algorithm

In this section we show how the accelerated gradient method can be used to distributively solve (3) by minimizing the negative dual function f. The accelerated proximal gradient method for problem (10) is defined by the following iteration as presented in [8, Algorithm 2] and [7, Eq. 4.1-4.3]

$$v^k = z^k + \frac{k-1}{k+2}(z^k - z^{k-1}) \tag{14}$$

$$z^{k+1} = \mathcal{P}_Z\left(v^k - \frac{1}{L}\nabla f(v^k)\right) \tag{15}$$

where $\mathcal{P}_Z$ is the Euclidean projection onto the set Z. Thus, the new iterate, z^{k+1}, is the previous iterate plus a step in the negative gradient direction projected onto the feasible set.

For reasons that will be revealed later we define the primal iteration $x^k := H^{-1}(-\mathcal{A}^T z^k - g)$. Using this definition, straightforward insertion of v^k into (11) gives

$$\nabla f(v^k) = -\mathcal{A}\left(x^k + \frac{k-1}{k+2}(x^k - x^{k-1})\right) + \mathcal{B}$$

By defining $\bar{x}^k = x^k + \frac{k-1}{k+2}(x^k - x^{k-1})$ and recalling the partition $z = [\lambda^T\ \mu^T\ \nu^T]^T$ and the definition (9) of the set Z, we find that (14)-(15) can be parallelized:

$$x^k = H^{-1}(-\mathcal{A}^T z^k - g) \tag{16}$$

$$\bar{x}^k = x^k + \frac{k-1}{k+2}(x^k - x^{k-1}) \tag{17}$$

$$\lambda_l^{k+1} = \lambda_l^k + \frac{k-1}{k+2}(\lambda_l^k - \lambda_l^{k-1}) + \frac{1}{L}(a_l^T\bar{x}^k - b_l) \tag{18}$$

$$\mu_l^{k+1} = \max\left\{0, \mu_l^k + \frac{k-1}{k+2}(\mu_l^k - \mu_l^{k-1}) + \frac{1}{L}(a_l^T\bar{x}^k - b_l)\right\} \tag{19}$$

$$\nu_r^{k+1} = \min\Big\{\gamma, \max\big[-\gamma, \nu_r^k + \frac{k-1}{k+2}(\nu_r^k - \nu_r^{k-1}) + \frac{1}{L}(P_r^T\bar{x}^k - p_r)\big]\Big\}. \tag{20}$$

From these iterations it is not clear that the algorithm is distributed. By partitioning the constraint matrix as

$$\mathcal{A} = [\mathcal{A}_1, \ldots, \mathcal{A}_M]$$

where each $\mathcal{A}_i = [a_{1i}, \ldots, a_{si}, P_{1i}, \ldots, P_{mi}]^T \in \mathbb{R}^{(s+m)\times n_i}$, and noting that H is block-diagonal, the local primal variables are updated according to

$$\begin{aligned} x_i^k &= H_i^{-1}\big(-\mathcal{A}_i^T z^k - g_i\big) \\ &= -H_i^{-1}\Big(g_i + \sum_{j\in\mathcal{M}_i}\Big[\sum_{l\in\mathcal{L}_j^1} a_{li}\lambda_l^k + \sum_{l\in\mathcal{L}_j^2} a_{li}\mu_l^k + \sum_{r\in\mathcal{R}_j} P_{ri}\nu_r^k\Big]\Big) \end{aligned} \tag{21}$$

Thus, each local primal update, x_i^k, can be computed after communication with neighbors $j \in \mathcal{M}_i$. Through (4)-(6) we note that the dual variable iterations can be updated after communication with neighbors $i \in \mathcal{N}_i$. We get the following distributed algorithm.

Algorithm 4.1. ***Distributed accelerated proximal gradient algorithm***

Initialize $\lambda^0 = \lambda^{-1}, \mu^0 = \mu^{-1}, \nu^0 = \nu^{-1}$ *and* $x^0 = x^{-1}$
In every node, i, the following computations are performed:
For $k \geq 0$

1. *Compute* x_i^k *according to* (21) *and set*

$$\bar{x}_i^k = x_i^k + \frac{k-1}{k+2}(x_i^k - x_i^{k-1})$$

2. *Send* $\bar{x}_i^k$ *to each* $j \in \mathcal{M}_i$*, receive* $\bar{x}_j^k$ *from each* $j \in \mathcal{N}_i$

3. *Compute* λ_l^{k+1} *according to* (18), (4) *for* $l \in \mathcal{L}_i^1$
   Compute μ_l^{k+1} *according to* (19), (5) *for* $l \in \mathcal{L}_i^2$
   Compute ν_l^{k+1} *according to* (20), (6) *for* $l \in \mathcal{R}_i$
4. *Send* $\{\lambda_l^{k+1}\}_{l\in\mathcal{L}_i^1}, \{\mu_l^{k+1}\}_{l\in\mathcal{L}_i^2}, \{\nu_r^{k+1}\}_{r\in\mathcal{R}_i}$ *to each* $j \in \mathcal{N}_i$*,*
   receive $\{\lambda_l^{k+1}\}_{l\in\mathcal{L}_j^1}, \{\mu_l^{k+1}\}_{l\in\mathcal{L}_j^2}$ *and* $\{\nu_r^{k+1}\}_{r\in\mathcal{R}_j}$ *from each* $j \in \mathcal{M}_i$

The convergence rates for the dual function f and the primal variables when running Algorithm 4.1 are stated in the following theorem.

Theorem 1. *Algorithm 4.1 has the following convergence rate properties:*

1. *Denote an optimizer of the dual problem* (10) *as* z^**. The convergence rate is:*

$$f(z^k) - f(z^*) \leq \frac{2L\|z^0 - z^*\|_2^2}{(k+1)^2}, \forall k \geq 1 \tag{22}$$

2. *Denote the unique optimizer of the primal problem as* x^**. The rate of convergence for the primal variable is*

$$\|x^k - x^*\|_2^2 \leq \frac{4\bar{\sigma}L\|z^0 - z^*\|_2^2}{\underline{\sigma}^2(k+1)^2}, \forall k \geq 1 \tag{23}$$

Proof. The derivation of Algorithm 4.1 shows that it is a distributed implementation of [8, Algorithm 2] and [7, Eq. 4.1-4.3] applied to minimize f. The convergence rate in argument 1 follows from [8, Proposition 2] and [7, Theorem 4.4].

For argument 2 we first show that $x^* = H^{-1}(-\mathcal{A}^T z^* - g)$. KKT-conditions [25, p. 244] imply that the primal optimal solution, x^*, and dual optimal solutions z^* must satisfy

$$0 = Hx^* + g + \mathcal{A}^T z^* \Leftrightarrow x^* = H^{-1}(-\mathcal{A}^T z^* - g)$$

since H is invertible. This leads to

$$\begin{aligned}\|x^k - x^*\|_2^2 &= \|H^{-1}(\mathcal{A}^T z^k - \mathcal{A}^T z^*)\|_2^2 \\ &\leq \|H^{-1}\|^2 \|\mathcal{A}^T z^k - \mathcal{A}^T z^*\|_2^2 \\ &\leq \frac{1}{\underline{\sigma}^2}\|\mathcal{A}^T z^k - \mathcal{A}^T z^*\|_2^2 \\ &\leq \frac{2\bar{\sigma}}{\underline{\sigma}^2}(f(z^k) - f(z^*)) \leq \frac{4\bar{\sigma}L\|z^0 - z^*\|_2^2}{\underline{\sigma}^2(k+1)^2}.\end{aligned}$$

where the second inequality is based on Remark 1, the third inequality on Proposition 2 and the final inequality is from (22). This completes the proof. □

4.1. Distributed step-size computations

To compute the step-size $\frac{1}{L}$ for the distributed algorithm, the Lipschitz constant $L = \|\mathcal{A}H^{-1}\mathcal{A}^T\|_2$ for ∇f is needed. However, this is the 2-norm of a matrix whose elements are spread over the different computational units, which may require non-negligible computational/communicational overhead to compute. We have in Proposition 1 shown the smallest Lipschitz constant to ∇f. However, any Lipschitz constant to ∇f can be used to choose the step-size and the results of Theorem 1 still hold. In this section we present two ways to, in distributed fashion, compute a Lipschitz constant.

We define $H_{\mathcal{A}} := \mathcal{A}H^{-1}\mathcal{A}^T$. The first method relies on the following matrix result [24, p. 313]

$$\|H_{\mathcal{A}}\|_2 \leq \|H_{\mathcal{A}}\|_F \triangleq \sqrt{\sum_p \sum_q [H_{\mathcal{A}}]_{pq}^2}$$

Let p correspond to some $l \in \mathcal{L}_i^1$ and q to some constraint k, then $[H_{\mathcal{A}}]_{pq} = a_l^T H^{-1} a_k = \sum_{j \in \mathcal{N}_i} a_{lj} H_j^{-1} a_{kj}$ can be computed in a distributed fashion. All local terms can be communicated to the other nodes to compute the norm.

Another alternative is to rely on the following result, [24, p. 313]

$$\|H_{\mathcal{A}}\|_2 \leq \sqrt{\|H_{\mathcal{A}}\|_1 \|H_{\mathcal{A}}\|_\infty} = \|H_{\mathcal{A}}\|_1 \triangleq \max_p \left(\sum_q |[H_{\mathcal{A}}]_{pq}| \right)$$

where the first equality holds since $H_{\mathcal{A}}$ is symmetric. As previously noted, the elements $[H_{\mathcal{A}}]_{pq}$ can be computed in a distributed fashion and communicated to the other nodes to compute the norm.

In the following section we will evaluate the algorithm with different step-sizes. We denote by $L_F = \|H_{\mathcal{A}}\|_F$ and $L_1 = \|H_{\mathcal{A}}\|_1$ and compare the performance using step-sizes $\frac{1}{L_F}$, $\frac{1}{L_1}$ to the optimal $\frac{1}{L}$.

5. Numerical Example

In this section we evaluate the performance of Algorithm 4.1. We compare the presented algorithm to state-of-the-art centralized optimization software for large-scale optimization implemented in C, namely CPLEX and MOSEK. We also evaluate the performance loss when using sub-optimal step-sizes. Our algorithm is implemented on a single processor to be able to compare execution times.

The comparison is made on 100 random optimization problems arising in distributed MPC. A batch of random stable controllable dynamical systems with random structure and random initial conditions are created. The sparsity fraction, i.e. the fraction of non-zero elements in the dynamics matrix and the input matrix, is chosen to be 0.1. We have random inequality constraints that are generated to guarantee a feasible solution and a 1-norm cost where the P-matrix and p-vector are randomly chosen. The quadratic cost matrices are chosen $Q = I$ and $R = I$. To summarize, we consider problems of the form

$$\begin{aligned} \min_{x,u} \quad & \left\{ \sum_{t=0}^{N-1} x(t)^T Q x(t) + u(t)^T R u(t) + \left\| P \begin{bmatrix} x(t) \\ u(t) \end{bmatrix} - p \right\|_1 \right\} \\ & x(t+1) = Ax(t) + Bu(t), \quad t = 0, \dots, N-2 \\ \text{s.t.} \quad & A_1 \begin{bmatrix} x(t) \\ u(t) \end{bmatrix} \leq b_1, \qquad t = 0, \dots, N-1 \\ & x(0) = x_0 \end{aligned} \tag{24}$$

Table 1 shows the numerical results obtained by a Linux PC with a 3 GHz Intel Core i7 processor and 4 GB memory.

Table 1: Algorithm comparison with 1-norm cost term and random state and input constraints. Algorithm 4.1 is implemented in MATLAB, while the others are implemented in C.

Alg.	vars./constr.	tol.	# iters		# exec (ms)	
			mean	max	mean	max
4.1 (L)	4320/3231	0.005	69.8	160	253	609
4.1 (L_1)	4320/3231	0.005	160	420	594	1532
4.1 (L_F)	4320/3231	0.005	248	640	934	2444
MOSEK	4320/3231	-	-	-	1945	2674
CPLEX	4320/3231	0.005	-	-	1663	2832
4.1 (L)	2160/1647	0.005	63.8	100	94	200
4.1 (L_1)	2160/1647	0.005	75.8	180	115	368
4.1 (L_F)	2160/1647	0.005	121	320	185	488
MOSEK	2160/1647	-	-	-	334	399
CPLEX	2160/1647	0.005	-	-	282	522

The first column specifies the algorithm used where 4.1 is supplemented with the step-size used. The second column specifies the number of variables and constraints in the optimization problems. In the third column we have information about the duality gap tolerance that is used as stopping condition in the algorithms (if possible to set). The two final columns present the results in terms of number of iterations and execution time. The difference between the upper and lower halves of the table is the size of the problems that are solved.

Table 1 reveals that Algorithm 4.1 performs better than CPLEX and MOSEK on these large-scale sparse problems despite the fact that CPLEX and MOSEK are implemented in C and Algorithm 4.1 is implemented in MATLAB. We also conclude that the choice of step-size in Algorithm 4.1 is important for performance reasons.

6. Conclusions

We have presented a distributed optimization algorithm for strongly convex optimization problems with sparse problem data. The algorithm is based on an accelerated gradient method that is applied to the dual problem. The algorithm was applied to large-scale sparse optimization problems originating from a distributed model predictive control formulation. Our algorithm performed better than state-of-the-art optimization software for large-scale sparse optimization, namely CPLEX and MOSEK, on these problems.

7. Acknowledgments

The second author would like to thank Quoc Tran Dinh for helpful discussions on the topic of this paper.

The second, third and fourth authors were supported by the European Union Seventh Framework STREP project "Hierarchical and distributed model predictive control (HD-MPC)", contract number INFSO-ICT-223854, and the European Union Seventh Framework Programme [FP7/2007-2013] under grant agreement no. 257462 HYCON2 Network of Excellence.

The first and last authors were supported by the Swedish Research Council through the Linnaeus center LCCC.

References

[1] D. P. Bertsekas, Nonlinear Programming, Athena Scientific, Belmont, MA, 2nd edition, 1999.

[2] Y. Nesterov, Introductory Lectures on Convex Optimization: A Basic Course (Applied Optimization), Springer Netherlands, 1 edition, 2004.

[3] A. Nemirovsky, D. Yudin, Informational Complexity and Efficient Methods for Solution of Convex Extremal Problems, Wiley, NewYork, NY, 1983.

[4] Y. Nesterov, A method of solving a convex programming problem with convergence rate O ($1/k^2$), Soviet Mathematics Doklady 27 (1983) 372–376.

[5] Y. Nesterov, On an approach to the construction of optimal methods of minimization of smooth convex functions, Ékonom. i. Mat. Metody 24 (1988) 509–517.

[6] Y. Nesterov, Smooth minimization of non-smooth functions, Math. Program. 103 (2005) 127–152.

[7] A. Beck, M. Teboulle, A fast iterative shrinkage-thresholding algorithm for linear inverse problems, SIAM J. Imaging Sciences 2 (2009) 183–202.

[8] P. Tseng, On accelerated proximal gradient methods for convex-concave optimization, 2008. Submitted to SIAM J. Optim. Available: `http://www.math.washington.edu/~tseng/papers/apgm.pdf`.

[9] K. Arrow, L. Hurwicz, H. Uzawa, Studies in Linear and Nonlinear Programming, Stanford University Press, 1958.

[10] G. Danzig, P. Wolfe, The decomposition algorithm for linear programming, Econometrica 4 (1961) 767–778.

[11] W. Findeisen, Control and Coordination in Hierarchical Systems, International series on applied systems analysis, Wiley, 1980.

[12] M. D. Mesarovic, D. Macko, Y. Takahara, Theory of Hierarchical Multilevel Systems, Academic Press, New York, 1970.

[13] M. G. Singh, A. Titli, Systems: Decomposition, Optimisation, and Control, Pergamon, 1978.

[14] P. Carpentier, G. Cohen, Applied mathematics in water supply network management, Automatica 29 (1993) 1215 – 1250.

[15] J. Tsitsiklis, D. Bertsekas, M. Athans, Distributed asynchronous deterministic and stochastic gradient optimization algorithms, IEEE Transactions on Automatic Control 31 (1986) 803–812.

[16] M. D. Doan, T. Keviczky, B. De Schutter, An iterative scheme for distributed model predictive control using Fenchel's duality, Journal of Process Control 21 (2011) 746–755. Special Issue on Hierarchical and Distributed Model Predictive Control.

[17] M. D. Doan, T. Keviczky, I. Necoara, M. Diehl, B. De Schutter, A distributed version of Han's method for DMPC using local communications only, Control Engineering and Applied Informatics 11 (2009) 6–15.

[18] P. Giselsson, A. Rantzer, Distributed model predictive control with suboptimality and stability guarantees, in: Proceedings of the 49th Conference on Decision and Control, Atlanta, GA, pp. 7272 – 7277.

[19] R. Negenborn, B. De Schutter, J. Hellendoorn, Multi-agent model predictive control for transportation networks: Serial versus parallel schemes, Engineering Applications of Artificial Intelligence 21 (2008) 353–366.

[20] C. Savorgnan, C. Romani, A. Kozma, M. Diehl, Multiple shooting for distributed systems with applications in hydro electricity production, Journal of Process Control 21 (2011) 738 – 745.

[21] I. Necoara, J. Suykens, Application of a smoothing technique to decomposition in convex optimization, IEEE Transactions on Automatic Control 53 (2008) 2674 –2679.

[22] M. Kögel, R. Findeisen, A fast gradient method for embedded linear predictive control, in: Proceedings of the 18th IFAC World Congress, Milan, Italy, pp. 1362–1367.

[23] S. Richter, C. Jones, M. Morari, Real-time input-constrained MPC using fast gradient methods, in: Proceedings of the 48th Conference on Decision and Control, Shanghai, China, pp. 7387 – 7393.

[24] R. A. Horn, C. R. Johnson, Matrix Analysis, Cambridge University Press, 1990.

[25] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, New York, NY, 2004.

[26] K. Rockafellar, Convex Analysis, Princeton Univercity Press, Princeton, NJ, 1970.

[27] J. Gauvin, A necessary and sufficient regularity condition to have bounded multipliers in nonconvex programming, Mathematical Programming 12 (1977) 136–138.

8. Appendix

In this section we present the proofs for the propositions stating the properties of the dual function.

8.1. Proof for Proposition 1

For convenience we introduce $H_{\mathcal{A}} = \mathcal{A}H^{-1}\mathcal{A}^T$. We have

$$\|\nabla f(z_1) - \nabla f(z_2)\|_2 = \|H_{\mathcal{A}}(z_1 - z_2)\|_2 \leq \|H_{\mathcal{A}}\|_2\|z_1 - z_2\|_2$$

due to the Cauchy-Schwarz inequality. This shows that (12) holds. Next we show that $L = \|H_{\mathcal{A}}\|_2$ is the smallest Lipschitz constant on Z. From the definition (9) of Z we conclude that there exist $z_1 \in Z$ and $z_2 \in Z$ such that the difference $d_z = z_1 - z_2$ with $\|d_z\| = \epsilon$ is parallel to the eigen-vector $v_{\max}(H_{\mathcal{A}})$ corresponding to the largest eigen-value $\lambda_{\max}(H_{\mathcal{A}})$. By choosing $v_{\max}(H_{\mathcal{A}})$ such that $\|v_{\max}(H_{\mathcal{A}})\| = 1$ we get for some $z_1, z_2 \in Z$:

$$\begin{aligned}\|\nabla f(z_1) - \nabla f(z_2)\|_2 &= \|H_{\mathcal{A}}(z_1 - z_2)\|_2 = \|H_{\mathcal{A}}d_z\|_2 \\ &= \|H_{\mathcal{A}}v_{\max}(H_{\mathcal{A}})\epsilon\|_2 \\ &= \lambda_{\max}(H_{\mathcal{A}})\epsilon\|v_{\max}(H_{\mathcal{A}})\|_2 \\ &= \|H_{\mathcal{A}}\|_2\epsilon = \|H_{\mathcal{A}}\|_2\|d_z\| \\ &= \|H_{\mathcal{A}}\|_2\|z_1 - z_2\|\end{aligned}$$

This completes the proof. □

8.2. Proof for Proposition 2

Under Assumption 2 it is well known (cf. [26, Corollary 28.2.2]) that the set Z^* is non-empty. Further from [27] we know that Assumption 2 is equivalent to the Mangasarian-Fromovitz Constraint Qualification (MFCQ). In [27] it is shown that MFCQ is equivalent to Z^* being bounded. This shows argument 1.

To prove argument 2 we introduce $f_1(y) = \frac{1}{2}y^T H^{-1} y$, which gives $f(z) = f_1(\mathcal{A}^T z + g) + \mathcal{B}^T z$. From Remark 1 we know that $H^{-1} \succeq \frac{1}{\bar{\sigma}}I$ and hence that f_1 is strongly convex and satisfies (c.f. [2, Definition 2.1.2])

$$f_1(y_1) \geq f_1(y_2) + \langle \nabla f_1(y_2), y_1 - y_2\rangle + \frac{1}{2\bar{\sigma}}\|y_1 - y_2\|^2$$

We set $y_1 = \mathcal{A}^T z + g$ for any $z \in Z$ and $y_2 = \mathcal{A}^T z^* + g$ for any $z^* \in Z^*$. This gives

$$f(z) = f_1(\mathcal{A}^T z + g) + \mathcal{B}^T z$$

$$
\begin{aligned}
&\geq f_1(\mathcal{A}^T z^* + g) + \langle \nabla f_1(\mathcal{A}^T z^* + g), \mathcal{A}^T z + g - \mathcal{A}^T z^* - g \rangle \\
&\quad + \frac{1}{2\bar{\sigma}} \|\mathcal{A}^T z + g - \mathcal{A}^T z^* - g\|_2^2 + \mathcal{B}^T z + \mathcal{B}^T (z^* - z^*) \\
&= f(z^*) + \langle \mathcal{A} \nabla f_1(\mathcal{A}^T z^* + g) + \mathcal{B}, z - z^* \rangle + \frac{1}{2\bar{\sigma}} \|\mathcal{A}^T (z - z^*)\|_2^2 \\
&= f(z^*) + \langle \nabla f(z^*), z - z^* \rangle + \frac{1}{2\bar{\sigma}} \|\mathcal{A}^T (z - z^*)\|_2^2 \\
&\geq f(z^*) + \frac{1}{2\bar{\sigma}} \|\mathcal{A}^T (z - z^*)\|_2^2
\end{aligned}
$$

where the last inequality comes from the first-order optimality condition for convex functions (cf. [2, Theorem 2.2.5])

$$\langle \nabla f(z^*), z - z^* \rangle \geq 0.$$

for any $z \in Z$ and $z^* \in Z^*$. This completes the proof. □