Chapter 10
Rate Analysis of Inexact Dual Fast Gradient Method for Distributed MPC

I. Necoara

Abstract In this chapter we propose a dual decomposition method based on inexact dual gradient information and constraint tightening for solving distributed model predictive control (MPC) problems for network systems with state-input constraints. The coupling constraints are tightened and moved in the cost using the Lagrange multipliers. The dual problem is solved by a fast gradient method based on approximate gradients for which we prove sublinear rate of convergence. We also provide estimates on the primal and dual suboptimality of the generated approximate primal and dual solutions and we show that primal feasibility is ensured by our method. Our analysis relies on the Lipschitz property of the dual MPC function and inexact dual gradients. We obtain a distributed control strategy that has the following features: state and input constraints are satisfied, stability of the plant is guaranteed, whilst the number of iterations for the suboptimal solution can be precisely determined.

10.1 Introduction

Different problems from control and estimation can be addressed within the framework of network systems [11]. In particular, model predictive control (MPC) has become a popular advanced control technology implemented in network systems due to its ability to handle hard input and state constraints. Network systems are complex and large in dimension, whose structure may be hierarchical, multistage or dynamical and they have multiple decision-makers.

Decomposition methods represent a very powerful tool for solving distributed control and estimation problems in network systems. The basic idea of these methods is to decompose the original large optimization problem into smaller subproblems.

I. Necoara(✉)
Automation and Systems Engineering Department, University Politehnica Bucharest, Bucharest, Romania
e-mail: ion.necoara@acse.pub.ro

Decomposition methods can be divided in two main classes: primal decomposition and dual decomposition methods. In primal decomposition the optimization problem is solved using the original formulation and variables and the complicating constraints are handled via methods such as feasible directions, Gauss-Jacobi type and others [3, 6, 10, 11, 20]. In dual decomposition the original problem is rewritten using Lagrangian relaxation for the complicating constraints and the dual problem is solved with a Newton or (sub)gradient algorithm [1, 2, 5, 11, 12, 14, 18]. Dual decomposition methods based on subgradient iteration and averaging, that produce primal solutions in the limit, can be found in [7, 9]. Converge rate analysis for the dual subgradient method has been studied e.g. in [13], where the authors provide estimates of order $\mathcal{O}(1/\sqrt{k})$ for the approximate solutions and for feasibility violation. Thus, an important drawback of the dual decomposition methods consists of the fact that feasibility of the primal variables can be ensured only at optimality, which is usually impossible to attain in practice. However, in many applications, e.g. from control and estimation, the constraints can represent different requirements due to physical limitations of actuators, safety limits and operating conditions of the controlled plant. Therefore, any control or estimation scheme must ensure feasibility. Thus, we are interested in developing dual optimization algorithms which satisfy the requirement of feasibility.

One way to ensure feasibility of the primal variables in distributed MPC is through constraint tightening [5, 8, 17], or through distributed implementations of some classical methods, such as the method of feasible directions, Gauss-Jacobi type and others [3, 6, 20]. In [5] a dual distributed algorithm based on constraint tightening is presented for solving MPC problems for systems with coupled dynamics. The authors prove the convergence of the algorithm using the analysis of the dual subgradient method from [13], which has very slow convergence. In [8] the authors propose a decentralized MPC algorithm that uses the constraint tightening technique to achieve robustness while guaranteeing robust feasibility of the entire system. In [20] a cooperative based distributed MPC algorithm is proposed that converges to the centralized solution. In [3] a distributed algorithm based on the method of feasible directions is proposed that also converges to the centralized solution and guarantees primal feasibility. While most of the work cited above focuses on a primal approach, our work develops dual methods that ensure constraint feasibility, tackles more complex constraints and provides better estimates on suboptimality.

The main contribution in this chapter is to provide a novel distributed algorithm based on dual decomposition and constraint tightening for solving constrained MPC problems in network systems. The algorithm has better convergence rates than the algorithms in [5, 13] due to the fact that we exploit the Lipschitz property of the gradient of the dual MPC function. Further, we solve the inner problems only up to a certain accuracy by means of a parallel coordinate descent method that has linear rate of convergence. Even if we use inexact information of the gradient of the dual tightened function, after a finite number of iterations we are able to provide a primal feasible solution for our original MPC problem using averaging. We also derive estimates on dual and primal suboptimality of the generated approximate solutions for our MPC problem. Finally, we obtain a distributed MPC scheme that has the following features: state and input constraints are satisfied, stability of the plant is guaranteed, whilst the number of iterations for the suboptimal solution can be precisely determined.

The chapter is organized as follows. In Sect. 10.2 we introduce the MPC problem for network systems with state-input constraints. In Sect. 10.3 we develop an inexact dual fast gradient scheme for solving a tightened MPC problem and analyze its convergence. Section 10.4 shows how we can perform distributed computations in the MPC scheme. Its efficiency is demonstrated on a four-tank plant in Sect. 10.5.

Notation: For $u, v \in \mathbb{R}^n$ we denote the standard Euclidean inner product $\langle u, v\rangle = \sum_{i=1}^{n} u^i v^i$, the Euclidean norm $\|u\| = \sqrt{\langle u, u\rangle}$ and the projection onto $\mathbb{R}^n_+$ as $[u]^+$.

10.2 Distributed MPC for Network Systems

Many applications from control and estimation theory can be posed in the framework of optimization problem (10.6), e.g. the MPC problem for interconnected subsystems. We consider network systems which are comprised of M interconnected subsystems, whose dynamics are defined as:

$$x_i(k+1) = \sum_{j \in \mathcal{N}^i} A_{ij} x_j(k) + B_{ij} u_j(k), \tag{10.1}$$

where $x_i(k) \in \mathbb{R}^{n_{x_i}}$ and $u_i(k) \in \mathbb{R}^{n_{u_i}}$ represent the state and the input of ith subsystem at time k, $A_{ij} \in \mathbb{R}^{n_{x_i} \times n_{x_j}}$ and $B_{ij} \in \mathbb{R}^{n_{x_i} \times n_{u_j}}$ and $\mathcal{N}^i$ denotes the neighbors of the ith subsystem including i. In a particular case frequently found in literature [12, 20] the influence between neighboring subsystems is given only in terms of inputs:

$$x_i(k+1) = A_{ii} x_i(k) + \sum_{j \in \mathcal{N}^i} B_{ij} u_j(k). \tag{10.2}$$

We also impose local state and input constraints:

$$x_i(k) \in X_i, \quad u_i(k) \in U_i \quad \forall i = 1, \cdots, M, k \geq 0, \tag{10.3}$$

where $X_i \subseteq \mathbb{R}^{n_{x_i}}$ and $U^i \subseteq \mathbb{R}^{n_{u_i}}$ are simple convex sets. For the systems (10.1) or (10.2) and a prediction horizon of length N_p, we consider a convex cost function composed of a stage and a final cost for each subsystem i: $\sum_{l=0}^{N_\mathrm{p}-1} \ell_i(x_i(l), u_i(l)) + \ell_i^\mathrm{f}(x_i(N_\mathrm{p}))$. The centralized MPC problem for (10.1) for a given initial state x is formulated as:

$$J^*(x) = \min_{x_i(l)\in X_i,\ u_i(l)\in U_i} \sum_{i=1}^{M} \sum_{l=0}^{N_\mathrm{p}-1} \ell_i(x_i(l), u_i(l)) + \ell_i^\mathrm{f}(x_i(N_\mathrm{p})) \tag{10.4}$$
$$\text{s.t: } x_i(l+1) = \sum_{j\in\mathcal{N}^i} A_{ij}x_j(l) + B_{ij}u_j(l),\ x_i(0) = x_i,\ x_i(N_\mathrm{p}) \in X_i^\mathrm{f} \quad \forall i,$$

where X_i^f are terminal sets chosen under some appropriate conditions to ensure stability of the MPC scheme (see e.g. [19]). For the input trajectory of subsystem i and the overall input trajectory we use the notation (here $n_i = N_\mathrm{p} n_{u_i}$ and $n = \sum_{i=1}^{M} n_i$):

$$\mathbf{u}_i = \left[u_i^T(0) \cdots u_i^T(N_\mathrm{p}-1)\right]^T \in \mathbb{R}^{n_i}, \quad \mathbf{u} = \left[\mathbf{u}_1^T \cdots \mathbf{u}_M^T\right]^T \in \mathbb{R}^n.$$

By eliminating the states from dynamics (10.1) and assuming that X_i and X_i^f are polyhedral sets for all i, problem (10.4) can be expressed as a large-scale convex problem:

$$J^*(x) = \min_{\mathbf{u}_1\in\mathbf{U}_1,\cdots,\mathbf{u}_M\in\mathbf{U}_M} \{J(x, \mathbf{u}_1, \cdots, \mathbf{u}_M) : \quad \mathbf{G}\mathbf{u} + \mathbf{E}x + \mathbf{g} \le 0\} \tag{10.5}$$

where the convex sets $\mathbf{U}_i$ are the Cartesian product of the sets U_i for N_p times and the inequalities $\mathbf{G}\mathbf{u} + \mathbf{E}x + \mathbf{g} \le 0$ are obtained by eliminating the states from the constraints $x_i(l) \in X_i$ and $x_i(N_\mathrm{p}) \in X_i^\mathrm{f}$ for all l and i.

10.3 Solving the Dual MPC Problem Using an Inexact Fast Gradient Method and Tightened Constraints

In this section we present briefly an optimization algorithm for solving distributively the MPC problem (10.5). For brevity, we remove the dependence of the cost and constraints in (10.5) on the initial state x and consider the smooth convex problem:

$$F^* = \min_{\mathbf{u}\in\mathbf{U}} \{F(\mathbf{u}) : \quad \mathbf{G}\mathbf{u} + \mathbf{g} \le 0\}, \tag{10.6}$$

where $F : \mathbb{R}^n \to \mathbb{R}$ is a convex function, $\mathbf{U} \subseteq \mathbb{R}^n$ is a convex set and $\mathbf{G} \in \mathbb{R}^{m\times n}$, $\mathbf{g} \in \mathbb{R}^m$. We assume that projection on the set defined by the coupling constraints (called also *complicating constraints*) $\mathbf{G}\mathbf{u} + \mathbf{g} \le 0$ is hard to compute, but the local constraints are in the form of a Cartesian product $\mathbf{U} = \mathbf{U}_1 \times \cdots \times \mathbf{U}_M$, with $\mathbf{U}_i \subseteq \mathbb{R}^{n_i}$ being simple sets, i.e. the projection on these sets can be computed efficiently (e.g. box sets or entire space $\mathbb{R}^{n_i}$). We define the partition of identity matrix: $I_n = [E_1 \cdots E_M] \in \mathbb{R}^{n\times n}$, where $E_i \in \mathbb{R}^{n\times n_i}$ for all $i = 1, \cdots, M$. Thus, $\mathbf{u}$ can be represented as: $\mathbf{u} = \sum_{i=1}^{M} E^i \mathbf{u}_i$. We define the partial gradient of F at $\mathbf{u}$, denoted $\nabla_i F(\mathbf{u}) \in \mathbb{R}^{n_i}$, as: $\nabla_i F(\mathbf{u}) = (E^i)^T \nabla F(\mathbf{u})$ for all i. For simplicity we use the short notation: $h(\mathbf{u}) = \mathbf{G}\mathbf{u} + \mathbf{g}$. The following assumptions are considered in this paper:

Assumption 10.1 *(i) Function F is σ_F-strongly convex w.r.t $\|\cdot\|$.*

(ii) The gradient of F is coordinate-wise Lipschitz continuous with constants $L_i > 0$, i.e: $\|\nabla_i F(\mathbf{u} + E_i d_i) - \nabla_i F(\mathbf{u})\| \leq L_i \|d_i\|$ for all $\mathbf{u} \in \mathbb{R}^n$, $d_i \in \mathbb{R}^{n_i}$.

Assumption 10.2 *Slater constraint qualification holds for (10.6), i.e. there exists vector $\tilde{\mathbf{u}} \in \mathbf{U}$ such that $\mathbf{G}\tilde{\mathbf{u}} + \mathbf{g} < 0$.*

In dual decomposition, instead of minimizing the primal function $F(\mathbf{u})$, one has to maximize the dual function $d(\lambda) = \min_{\mathbf{u}\in\mathbf{U}} L(\mathbf{u}, \lambda)$, where $L(\mathbf{u}, \lambda) = F(\mathbf{u}) + \langle \lambda, h(\mathbf{u})\rangle$ denotes the partial Lagrangian w.r.t. the complicating constraints $h(\mathbf{u}) \leq 0$. Since we assume the set $\mathbf{U}$ to be simple, any gradient or Newton based projection method for solving the inner subproblems, for a given λ, has cheap iterations. Moreover, based on Assumption 10.1 (i) the dual function $d(\lambda)$ has Lipschitz continuous gradient with constant $L_\text{d} = \|\mathbf{G}\|^2/\sigma_\text{F}$ (see e.g. [15]). As a consequence of Assumption 10.2, we also have that strong duality holds.

10.3.1 Tightening the Coupling Constraints

In many applications, e.g. the MPC problem from Sect. 10.2, the constraints may represent different requirements on physical limitations of actuators, safety limits and operating conditions of the controlled plant. Thus, ensuring the feasibility of the primal variables in (10.6), i.e. $\mathbf{u} \in \mathbf{U}$ and $\mathbf{G}\mathbf{u} + \mathbf{g} \leq 0$, becomes a prerequisite. However, dual decomposition methods can ensure these requirements only at optimality, which is usually impossible to attain in practice. Therefore, in our approach, instead of solving the original problem (10.6), we propose a first order optimization algorithm based on dual decomposition for a tightened problem (see also [5] for a similar approach where the tightened dual problem is solved using a subgradient algorithm). We introduce the following tightened problem associated with original problem (10.6):

$$F^*_{\epsilon_c} = \min_{\mathbf{u}\in\mathbf{U}} \{F(\mathbf{u}) : \quad \mathbf{G}\mathbf{u} + \mathbf{g} + \epsilon_c \mathbf{e} \leq 0\}, \tag{10.7}$$

where $\mathbf{e}$ denotes the vector with all entries 1 and

$$0 \leq \epsilon_\text{c} < \min\{-(\mathbf{G}\tilde{\mathbf{u}} + \mathbf{g})_1, \cdots, -(\mathbf{G}\tilde{\mathbf{u}} + \mathbf{g})_m\}, \tag{10.8}$$

where $\tilde{\mathbf{u}}$ is a Slater vector for (10.6) as in Assumption 10.2. Note that for this choice of ϵ_c, we have that $\tilde{\mathbf{u}}$ is also a Slater vector for problem (10.7), so that Assumption 10.2 remains valid for this problem. Our goal is to solve the optimization problem (10.7) using dual decomposition. In order to update the dual variables we use an inexact dual fast gradient method. An important feature of our algorithm consists of the fact that even if we use inexact information of the gradient of the dual function, we are still able to compute a sequence of primal variables which, after a certain number of outer iterations, become feasible and ϵ_out-optimal for our original problem (10.6).

Recall that we use the short notation $h(\mathbf{u}) = \mathbf{G}\mathbf{u}+\mathbf{g}$. We assume that projection on $\mathbf{U}_i$ is simple but the projection on the set defined by the tightened coupling constraints $h(\mathbf{u}) + \epsilon_{\mathrm{c}}\mathbf{e} \leq 0$ is hard to compute. Therefore, we remove the tightened constraints in the cost via Lagrange multipliers and define the dual function:

$$d_{\epsilon_{\mathrm{c}}}(\lambda) = \min_{\mathbf{u}\in\mathbf{U}} L_{\epsilon_{\mathrm{c}}}(\mathbf{u}, \lambda), \tag{10.9}$$

where $L_{\epsilon_{\mathrm{c}}}(\mathbf{u}, \lambda) = F(\mathbf{u}) + \langle \lambda, h(\mathbf{u}) + \epsilon_c \mathbf{e}\rangle$. Further, the gradient of the dual function $d_{\epsilon_{\mathrm{c}}}(\lambda)$ is Lipschitz continuous, with the same constant L_{d} as for the original dual function for (10.6), and is given by [2, 15]: $\nabla d_{\epsilon_{\mathrm{c}}}(\lambda) = h(\mathbf{u}(\lambda)) + \epsilon_{\mathrm{c}}\mathbf{e}$, where $\mathbf{u}(\lambda)$ is the optimal solution of the *inner problem*:

$$\mathbf{u}(\lambda) = \arg\min_{\mathbf{u}\in\mathbf{U}} L_{\epsilon_{\mathrm{c}}}(\mathbf{u}, \lambda). \tag{10.10}$$

Under strong duality we have for the *outer problem*:

$$F^*_{\epsilon_c} = \max_{\lambda\geq 0} d_{\epsilon_{\mathrm{c}}}(\lambda). \tag{10.11}$$

Remark 10.1 If Assumption 10.2 holds for (10.7) and $F^*_{\epsilon_{\mathrm{c}}}$ is finite, then from [2] it follows that the set of optimal Lagrange multipliers for the inequalities $h(\mathbf{u}) + \mathbf{e} \leq 0$ is bounded. Therefore, problem (10.11) can be written equivalently as: $F^*_{\epsilon_c} = \max_{\lambda\in Q} d_{\epsilon_{\mathrm{c}}}(\lambda)$, where

$$Q = \{\lambda \in \mathbb{R}^m : \lambda \geq 0,\ \|\lambda\| \leq R\}, \tag{10.12}$$

for some $R > 0$ such that $\lambda^* \in Q$, where λ^* denotes an optimal solution of (10.11). □

Since we cannot usually solve the inner optimization problem (10.10) exactly, but with some accuracy ϵ_{in} and obtaining an approximate solution $\bar{\mathbf{u}}(\lambda)$, we do not have available an exact gradient of the dual function in λ, but only an approximation:

$$\bar{\nabla} d_{\epsilon_{\mathrm{c}}}(\lambda) = h(\bar{\mathbf{u}}(\lambda)) + \epsilon_c \mathbf{e}.$$

If we assume that $\bar{\mathbf{u}}(\lambda)$ is computed such that the following ϵ_{in}-optimality holds:

$$\bar{\mathbf{u}}(\lambda) \in \mathbf{U},\quad L_{\epsilon_{\mathrm{c}}}(\bar{\mathbf{u}}(\lambda), \lambda) - L_{\epsilon_{\mathrm{c}}}(\mathbf{u}(\lambda), \lambda) \leq \epsilon_{\mathrm{in}}/2, \tag{10.13}$$

then next lemma gives bounds for the dual function $d_{\epsilon_{\mathrm{c}}}(\lambda)$ (see Sect. 3.1 in [4]):

Lemma 10.1 [4] *Let Assumptions 10.1 and 10.2 hold and for a given λ let $\bar{\mathbf{u}}(\lambda)$ be computed as in* (10.13). *Then, the following inequalities are valid for all $\tilde{\lambda} \in \mathbb{R}^m_+$:*

Algorithm 10.1 (IDFG)(λ^0)

```
Given $\lambda^0 \in \mathbb{R}^m_+$, for $p \geq 0$ compute:
1: $\bar{\mathbf{u}}^p \approx \arg\min_{\mathbf{u}\in\mathbf{U}} L_{\epsilon_c}(\mathbf{u}, \lambda^p)$ such that (10.13) holds
2: $\mu^p = \left[\lambda^p + \frac{1}{L_d}\bar{\nabla} d_{\epsilon_c}(\lambda^p)\right]^+$ and $\eta^p = \left[\lambda^0 + \frac{1}{L_d}\sum_{s=0}^{p}\frac{s+1}{2}\bar{\nabla} d_{\epsilon_c}(\lambda^s)\right]^+$
3: $\lambda^{p+1} = \frac{p+1}{p+3}\mu^p + \frac{2}{p+3}\eta^p$.
```

$$0 \geq d_{\epsilon_c}(\tilde{\lambda}) - [L_{\epsilon_c}(\bar{\mathbf{u}}(\lambda), \lambda) + \langle \bar{\nabla} d_{\epsilon_c}(\lambda), \tilde{\lambda} - \lambda\rangle] \geq -L_d\|\tilde{\lambda} - \lambda\|^2 - \epsilon_{in}. \quad (10.14)$$

Our goal is to solve the dual problem (10.11) using a fast gradient scheme with accuracy ϵ_{out}. We obtain such accuracy ϵ_{out} after p_{out} iterations and after which we construct a primal estimate $\hat{\mathbf{u}}^{p_{\text{out}}}$. We want to guarantee for this estimate primal feasibility and primal suboptimality of order ϵ_{out} for problem (10.6):

$$\hat{\mathbf{u}}^{p_{\text{out}}} \in \mathbf{U}, \quad \mathbf{G}\hat{\mathbf{u}}^{p_{\text{out}}} + \mathbf{g} \leq 0 \ \text{ and } \ F(\hat{\mathbf{u}}^{p_{\text{out}}}) - F^* \leq \mathcal{O}(\epsilon_{\text{out}}).$$

10.3.2 Inexact Dual Fast Gradient Method for Solving Outer Problem

In this section we discuss an inexact dual fast gradient scheme for updating λ. The algorithm was proposed by Nesterov [15] and applied further in [12] for solving dual problems with exact gradient information. The scheme defines three sequences $(\lambda^p, \mu^p, \eta^p)_{p\geq 0}$ for the dual variables, see Algorithm 10.1.

Based on Theorem 1 in [4], which is an extension of the results in [12, 15] to the inexact case, we have the following result:

Lemma 10.2 [4, 12] *If Assumptions 10.1 and 10.2 hold and sequences* $(\bar{\mathbf{u}}^p, \lambda^p, \mu^p, \eta^p)_{p\geq 0}$ *are generated by Algorithm 10.1, then for all* $p \geq 0$ *we have:*

$$\frac{(p+1)(p+2)}{4} d_{\epsilon_c}(\mu^p) \geq \max_{\lambda\geq 0} -L_d\|\lambda - \lambda^0\|^2 + \sum_{s=0}^{p}\frac{s+1}{2}\left[L_{\epsilon_c}(\bar{\mathbf{u}}^s, \lambda^s) + \langle\bar{\nabla} d_{\epsilon_c}(\lambda^s), \lambda - \lambda^s\rangle\right]$$
$$- \frac{(p+1)(p+2)(p+3)}{12}\epsilon_{in} \ \forall\lambda \in \mathbb{R}^m_+. \quad (10.15)$$

The next theorem provides estimates on the dual suboptimality of the generated approximate dual solutions of **(IDFG)**:

Theorem 10.1 *Let Assumptions 10.1 and 10.2 hold and the sequences* $(\bar{\mathbf{u}}^p, \lambda^p, \mu^p, \eta^p)_{p\geq 0}$ *be generated by Algorithm 10.1 with* $\lambda^0 = 0$. *Then, an estimate on dual suboptimality for the original problem* (10.6) *is given by:*

$$F^* - d_{\epsilon_c}(\mu^p) \leq \frac{4L_d R^2}{(p+1)^2} + \frac{p+3}{3}\epsilon_{in}. \tag{10.16}$$

Proof Using the first inequality from (10.14) in (10.15) we get:

$$\frac{(p+1)(p+2)}{4} d_{\epsilon_c}(\mu^p) \geq - L_d\|\lambda^*\|^2 + \sum_{s=0}^{p} \frac{s+1}{2} d_{\epsilon_c}(\lambda^*) - \frac{(p+1)(p+2)(p+3)}{12}\epsilon_{in}.$$

Dividing now both sides of the previous inequality by $\frac{(p+1)(p+2)}{4}$, using (10.12), rearranging the terms and taking into account that $d_{\epsilon_c}(\lambda^*) = F^*_{\epsilon_c} \geq F^*$ and $(p+1)^2 \leq (p+1)(p+2)$ we obtain (10.16). □

We are interested now in finding estimates on primal suboptimality and primal infeasibility for our original problem (10.6). For this purpose we define the following average sequence for the primal variables:

$$\bar{\mathbf{u}}^p = \sum_{s=0}^{p} \frac{2(s+1)}{(p+1)(p+2)} \bar{\mathbf{u}}^s. \tag{10.17}$$

The next theorems give estimates on primal suboptimality and infeasibility, whose proofs can be found in Appendix (Sect. 10.7).

Theorem 10.2 *Assume the conditions from Theorem* 10.1 *hold and let* $\bar{\mathbf{u}}^p$ *be given by* (10.17). *Then, the following estimate on primal suboptimality for* (10.6) *can be derived:*

$$F(\bar{\mathbf{u}}^p) - F^* \leq \sqrt{m} R \epsilon_c + \frac{p+3}{3}\epsilon_{in}. \tag{10.18}$$

Theorem 10.3 *Under the assumptions of Theorem 10.2, an estimate for primal infeasibility is given by: if we define* $v(p, \epsilon_{in}) = \frac{8L_d R\left(1+\sqrt{1+\frac{(p+1)^2(p+3)}{12L_d R^2}\epsilon_{in}}\right)}{(p+1)^2}$*, then*

$$\|[\mathbf{G}\bar{\mathbf{u}}^p + \mathbf{g} + \epsilon_c \mathbf{e}]^+\| \leq v(p, \epsilon_{in}). \tag{10.19}$$

Assume now that we fix the outer accuracy ϵ_{out} to a desired value such that:

$$\epsilon_{\text{out}} \leq 2\sqrt{m} R \min_{j=1,\cdots,m} \{-(\mathbf{G}\tilde{\mathbf{u}} + \mathbf{g})_j\},$$

where $\tilde{\mathbf{u}}$ is a Slater vector for (10.6) (see Assumption 10.2). We are now interested in finding the number of outer iterations p_{out} and a relation between ϵ_{out}, ϵ_{in} and ϵ_c such that primal feasibility holds and primal suboptimality will be less than ϵ_{out}:

$$\hat{\mathbf{u}}^{p_{\text{out}}} \in \mathbf{U}, \quad \mathbf{G}\hat{\mathbf{u}}^{p_{\text{out}}} + \mathbf{g} \leq 0 \ \text{ and } \ F(\hat{\mathbf{u}}^{p_{\text{out}}}) - F^* \leq \epsilon_{\text{out}}. \tag{10.20}$$

From (10.18), we can take for example:

$$\epsilon_{\rm c} = \frac{\epsilon_{\rm out}}{2\sqrt{m}R} \quad \text{and} \quad \epsilon_{\rm in} = \frac{3\epsilon_{\rm out}}{2(p_{\rm out}+3)}. \tag{10.21}$$

Using Theorem 10.2 we see that for this choice of $\epsilon_{\rm c}$ and $\epsilon_{\rm in}$ we ensure the second condition in (10.20), i.e. $\epsilon_{\rm out}$-optimality for original problem (10.6). To get primal feasibility we impose the condition $v(p_{\rm out}, \epsilon_{\rm in}) \le \epsilon_{\rm c}$, from which we obtain e.g. $(p_{\rm out}+1)^2 \ge \frac{16\sqrt{m}L_{\rm d}R^2}{\epsilon_{\rm out}}$ and thus we can take:

$$p_{\rm out} = \left\lfloor 4R\sqrt{\frac{\sqrt{m}L_{\rm d}}{\epsilon_{\rm out}}} \right\rfloor.$$

It follows immediately that for this choice of $\epsilon_{\rm c}$, $\epsilon_{\rm in}$ and $p_{\rm out}$ the approximate primal solution produced for our original problem (10.6) is feasible (i.e. $\hat{\mathbf{u}}^{p_{\rm out}} \in \mathbf{U}$ and $\mathbf{G}\hat{\mathbf{u}}^{p_{\rm out}} + \mathbf{g} \le 0$) and $\epsilon_{\rm out}$-optimal (i.e. $F(\hat{\mathbf{u}}^{p_{\rm out}}) - F^* \le \epsilon_{\rm out}$) and thus (10.20) holds.

Note that all the results derived in this section hold also for $\lambda^0 \neq 0$, but in this case the formulas are more cumbersome. Further, our results hold also in the case when we solve the inner problems exactly, i.e. $\epsilon_{\rm in} = 0$ in (10.13), or when $\mathbf{U} = \mathbb{R}^n$, i.e. the inner problems are unconstrained.

10.3.3 Parallel Coordinate Descent Method for Solving Inner Problem

In this section, due to space limitations, we present briefly a block-coordinate descent algorithm which permits solving in parallel, for a fixed λ^p, the inner problem (10.9) approximately: $\min\limits_{\mathbf{u}_1\in\mathbf{U}_1,\cdots,\mathbf{u}_M\in\mathbf{U}_M} L_{\epsilon_{\rm c}}(\mathbf{u}, \lambda^p)$. Based on Assumption 10.1 it follows that $L_{\epsilon_{\rm c}}(\mathbf{u}, \lambda) = F(\mathbf{u}) + \langle \lambda, \mathbf{G}\mathbf{u} + \mathbf{g} + \epsilon_{\rm c}\mathbf{e}\rangle$ is strongly convex and with coordinate-wise Lipschitz continuous gradient in the first variable. Since the algorithm can be applied to a larger class of problems, which also includes our problem (10.9), we consider the more general problem:

$$f^* = \min_{\mathbf{u}_1\in\mathbf{U}_1,\cdots,\ \mathbf{u}_M\in\mathbf{U}_1} f(\mathbf{u}), \tag{10.22}$$

where f is convex and satisfies Assumption 10.1 (i.e. it is $\sigma_{\rm F}$-strongly convex and has L_i-coordinate-wise Lipschitz continuous gradient) and $\mathbf{U}_i \subseteq \mathbb{R}^{n_i}$ are simple, convex sets (e.g. box sets, entire space $\mathbb{R}^{n_i}$, etc). There exist many parallel algorithms in the literature for solving the optimization problem (10.22): e.g. Jacobi-type algorithms [2, 20]. However, the rate of convergence for these algorithms is guaranteed under more conservative assumptions than the ones required for the parallel coordinate descent method proposed in this section (see [10] for more details).

Algorithm 10.2 (PCD)($\mathbf{u}^0$)

```
Given $\mathbf{u}^0$, for $q \geq 0$ do:
1: compute in parallel $v_i(\mathbf{u}^q)$ for $i = 1, \ldots, M$
2: update: $\mathbf{u}^{q+1} = \sum_{i=1}^{M} \frac{1}{M} w_i^+(\mathbf{u}^q)$.
```

Let us define the constrained coordinate update for our inner algorithm:

$$v_i(\mathbf{u}) = \arg\min_{v_i \in \mathbf{U}_i} \langle \nabla_i f(\mathbf{u}), v_i - \mathbf{u}_i \rangle + \frac{L_i}{2} \|v_i - \mathbf{u}_i\|^2$$
$$w_i^+(\mathbf{u}) = \mathbf{u} + E_i(v_i(\mathbf{u}) - \mathbf{u}_i) \;\; \forall i = 1, \cdots, M.$$

Algorithm 10.2 shows our *Parallel Coordinate Descent Method*, resembling the method in [20] but with a simpler iteration and guaranteed rate of convergence and which can also be viewed as a parallel version of the block-coordinate descent method from [16].

We can easily show that Algorithm 10.2 decreases the objective function at each iteration: $f(\mathbf{u}^{q+1}) \leq f(\mathbf{u}^q)$ for all $q \geq 0$ (see [10] for more properties of this algorithm). Note that if the sets $\mathbf{U}_i$ are simple (e.g. boxes) the projection on $\mathbf{U}_i$ can be easy (for boxes in $\mathcal{O}(n_i)$ operations) and if f has cheap coordinate derivatives (e.g. quadratic function with sparse Hessian), then the cost of computing $\nabla_i f(\mathbf{u})$ is less than $\mathcal{O}(n \cdot n_i)$. Thus, for quadratic problems the worst case complexity per iteration of our method is $\mathcal{O}(n^2)$. Note that the complexity per iteration of the method from [20] is at least $\mathcal{O}(n^2 + \sum_{i=1}^{M} n_i^3)$, provided that the local quadratic subproblems are solved with an interior point solver. The next theorem provides the convergence rate of Algorithm 10.2:

Theorem 10.4 [10] *If function f has L_i-coordinate-wise Lipschitz continuous gradient and is also σ_F-strongly convex, then the Algorithm 10.2 has linear rate of convergence.*

10.4 A Distributed MPC Scheme Based on Local Information

In this section we discuss some technical aspects for the implementation of our inexact dual decomposition method in the case of MPC problem (10.4) and its equivalent form (10.5) presented in Sect. 10.2. Usually, in the linear MPC framework, the local stage and final cost are taken of the following quadratic form:

$$\ell_i(x_i, u_i) = \|x_i\|_{Q_i}^2 + \|u_i\|_{R_i}^2, \quad \ell_i^{\mathrm{f}}(x_i) = \|x_i\|_{P_i}^2,$$

where $\|x\|_{P_i}^2 = x^T P_i x$, the matrix $Q_i, P_i \in \mathbb{R}^{n_{x_i} \times n_{x_i}}$ are positive semidefinite, whilst matrices $R_i \in \mathbb{R}^{n_{u_i} \times n_{u_i}}$ are positive definite. We also assume that the local constraints sets U_i, X_i and the terminal sets X_i^{f} are polyhedral. In this particular case, the objective function in (10.5) is quadratically strongly convex, having the form:

$$F(\mathbf{u}) = J(x, \mathbf{u}) = 0.5\, \mathbf{u}^T \mathbf{Q}\mathbf{u} + (\mathbf{W}x + \mathbf{w})^T \mathbf{u},$$

where $\mathbf{Q}$ is positive definite due to the assumption that all R_i are positive definite. Since we assume that the sets U_i, X_i and X_{f} are polyhedral, then after eliminating the dynamics (10.1) or (10.2) all the complicating constraints are gathered in:

$$h(\mathbf{u}) = \mathbf{G}\mathbf{u} + \mathbf{E}x + \mathbf{g} \leq 0.$$

If the projection on the set U_i is difficult we also move the input constraints in the complicating constraints defined above. In this case $\mathbf{U}_i = \mathbb{R}^{n_i}$. Otherwise (i.e. the local sets U_i are simple, e.g. boxes or the entire space $\mathbb{R}^{n_{u_i}}$) the convex sets $\mathbf{U}_i = \prod_{l=1}^{N_{\mathrm{p}}} U_i$. Usually, for the dynamics (10.1) the corresponding matrices $\mathbf{Q}$ and $\mathbf{G}$ obtained after eliminating the states are dense and despite the fact that both Algorithms 10.1 and 10.2 can perform parallel computations (i.e. each subsystem needs to solve small local problems) we need all to all communication between subsystems. However, for the dynamics (10.2) the corresponding matrices $\mathbf{Q}$ and $\mathbf{G}$ are sparse and in this case in our Algorithms 10.1 and 10.2 we can perform distributed computations (i.e. the subsystems solve small local problems in parallel and they need to communicate only with their neighbors). Indeed, if the dynamics of the system are given by (10.2), then $x_i(l) = A_{ii}^l x_i(0) + \sum_{s=1}^{l} \sum_{j \in \mathcal{N}^i} A_{ii}^{s-1} B_{ij} u_j(l-s)$ and thus the matrices $\mathbf{Q}$ and $\mathbf{G}$ have a sparse structure (see e.g. [3, 20]). In particular, the complicating constraints have the following structure: for matrix $\mathbf{G}$ the (i, j) block matrices of $\mathbf{G}$, denoted G_{ij}, are zero for all $j \notin \mathcal{N}^i$ for a given subsystem i, while the matrix $\mathbf{E}$ is block diagonal. Further, if we define the neighborhood subsystems of a certain subsystem i as $\hat{\mathcal{N}}^i = \mathcal{N}^i \cup \{l : \; l \in \mathcal{N}^j,\, j \in \bar{\mathcal{N}}^i\}$, where $\bar{\mathcal{N}}^i = \{j : \; i \in \mathcal{N}^j\}$, then the matrix $\mathbf{Q}$ has all the block matrices $\mathbf{Q}_{ij} = 0$ for all $j \notin \hat{\mathcal{N}}^i$ and the matrix $\mathbf{W}$ has all the block matrices $\mathbf{W}_{ij} = 0$ for all $j \notin \bar{\mathcal{N}}^i$, for any given subsystem i. Thus, the ith block components of both $\bar{\nabla} d_{\epsilon_{\mathrm{c}}}$ and $\nabla L_{\epsilon_{\mathrm{c}}}(\mathbf{u}, \lambda)$ can be computed using local information:

$$\bar{\nabla}_i d_{\epsilon_{\mathrm{c}}}(\lambda) = \sum_{j \in \mathcal{N}^i} \mathbf{G}_{ij}\mathbf{u}_j + \mathbf{E}_{ii} x_i + \mathbf{g}_i + \epsilon_{\mathrm{c}} \mathbf{e}, \tag{10.23}$$

$$\nabla_i L_{\epsilon_{\mathrm{c}}}(\mathbf{u}, \lambda) = \sum_{j \in \hat{\mathcal{N}}^i} \mathbf{Q}_{ij}\mathbf{u}_j + \sum_{j \in \bar{\mathcal{N}}^i} \left(\mathbf{W}_{ij} x_j + \mathbf{G}_{ij}^T \lambda_j \right) + \mathbf{w}_i. \tag{10.24}$$

Note that in the Algorithm 10.2 the only parameters that we need to compute are the Lipschitz constants L_i. However, in the MPC problem L_i does not depend on the initial state x and can be computed locally by each subsystem as: $L_i = \lambda_{\max}(\mathbf{Q}_{ii})$. From the previous discussion it follows immediately that the iterations of Algorithm 10.2 can be performed in parallel using distributed computations (see (10.24)).

Algorithm 10.3 Distributed MPC scheme $(x, \lambda^0, \mathbf{u}^0)$

```
Given $x$ and accuracy $\epsilon_{\text{out}}$ choose $\lambda^0 \in \mathbb{R}^p_+$ and $\mathbf{u}^0 \in \mathbf{U}$.
Write MPC problem (10.4) into optimization problem (10.5).

1: *Outer Iteration:* for $0 \leq p \leq p_{\text{out}}$ do
      *Inner iteration*: given $\mathbf{u}_i^0$ $\forall i$ and $\lambda^p$, for $q \geq 0$ repeat
   • Each subsystem $i$ computes in parallel $\nabla_i L_{\epsilon_c}(\mathbf{u}^q, \lambda^p)$ as in (10.24)
   • Each subsystem $i$ updates in parallel $\mathbf{u}_i^{q+1}$ using Algorithm 10.2 and $q \leftarrow q+1$.
   • Until $\bar{\mathbf{u}}^q = [\mathbf{u}_1^q, \cdots \mathbf{u}_M^q] \approx \arg\min_{\mathbf{u}\in\mathbf{U}} L_{\epsilon_c}(\mathbf{u}, \lambda^k)$ such that (10.13) holds
   • Define $\mathbf{u}^0 = \bar{\mathbf{u}}^q$ and compute $\hat{\mathbf{u}}^p$ as in (10.17)
2: Each subsystem $i$ computes in parallel $\bar{\nabla}_i d_{\epsilon_c}(\lambda^p)$ as in (10.23).
3: Each subsystem $i$ computes in parallel $(\mu_i^p, \eta_i^p, \lambda_i^{p+1})$ using Algorithm 10.1, $p \leftarrow p+1$.
```

Since the Algorithm 10.1 uses only first order information, we can observe that once $\bar{\nabla}_i d_{\epsilon_c}(\lambda)$ has been computed distributively, as proved in (10.23), all the computations for updating $(\lambda^p, \mu^p, \eta^p)$ can be done in parallel distributively, due to the fact that we have to do only vector operations. However, in the scheme 10.1 all subsystems need to know the global Lipschitz constant $L_{\text{d}} = \frac{\|\mathbf{G}\|^2}{\sigma_{\text{F}}} = \frac{\|\mathbf{G}\|^2}{\lambda_{\min}(\mathbf{Q})}$ that cannot be computed distributively. In practice, a good upper bound on $\frac{\|\mathbf{G}\|^2}{\lambda_{\min}(\mathbf{Q})}$ is sufficient. Note that we do not need to compute a Slater vector $\tilde{\mathbf{u}}$ for (10.4) as long as the MPC problem is well possed and the desired accuracy ϵ_{out} is sufficiently small such that $\epsilon_{\text{out}} \leq 2\sqrt{m}R \min_{j=1,\cdots,m}\{-(\mathbf{G}\tilde{\mathbf{u}} + \mathbf{E}x + \mathbf{g})_j\}$ for all x in some region of attraction $X_{N_{\text{p}}}$. In Algorithm 10.1 another global constant that has to be computed offline is an upper bound on the norm of the optimal multiplier, R_{d}, for any $x \in X_{N_{\text{p}}}$. Note that practical ways to compute an upper bound R_{d} can be found in [13, 18]. Once the desired accuracy was chosen and an upper bound R_{d} was computed offline, then we can take $\epsilon_{\text{c}} = \frac{\epsilon_{\text{out}}}{2\sqrt{m}R}$, as proven in previous sections.

From previous discussion we can conclude that the sequences $\left(\hat{\mathbf{u}}^p, \lambda^p, \mu^p, \eta^p\right)_{p\geq 0}$ generated by the Algorithms 10.1 and 10.2 can be computed in parallel and their updates can be performed distributively using local information, provided that dynamics (10.2) are considered and that good approximations for L_{d} and R_{d} can be computed offline. Closed loop stability for our MPC scheme as given as Algorithm 10.3 can be ensured by choosing the terminal costs ℓ_i^{f} and the terminal sets X_i^{f} for all i appropriately [19].

10.5 Numerical Example

To demonstrate the applicability of the new algorithms we consider the MPC problem for a 4-tank process, whose objective is to control the level of water in each tank. We obtain a continuous state-space model by linearizing the nonlinear model (see e.g. [1]) at an operating point given by $h_i^{\mathrm{e}}, \gamma_1^{\mathrm{e}}, \gamma_2^{\mathrm{e}}$ (where $h_1^{\mathrm{e}} = 0.19\,\mathrm{m}$, $h_2^{\mathrm{e}} = 0.13\,\mathrm{m}$, $h_3^{\mathrm{e}} = 0.23\,\mathrm{m}$, $h_4^{\mathrm{e}} = 0.09\,\mathrm{m}$ denote the water levels and $\gamma_1^{\mathrm{e}} = 0.58$, $\gamma_2^{\mathrm{e}} = 0.54$ represent the valve ratios) and the maximum inflows from the pumps $q_1^{max} = 0.39$ m^3/h, $q_2^{max} = 0.39$ m^3/h, with the deviation variables $x_i = h_i - h_i^{\mathrm{e}}$, $u_1 = \gamma_1 - \gamma_1^{\mathrm{e}}$, $u_2 = \gamma_2 - \gamma_2^{\mathrm{e}}$:

$$\frac{dx}{dt} = \begin{bmatrix} -\frac{1}{\tau_1} & 0 & 0 & \frac{1}{\tau_4} \\ 0 & -\frac{1}{\tau_2} & \frac{1}{\tau_3} & 0 \\ 0 & 0 & -\frac{1}{\tau_3} & 0 \\ 0 & 0 & 0 & -\frac{1}{\tau_4} \end{bmatrix} x + \begin{bmatrix} \frac{q_1^{max}}{S} & 0 \\ 0 & \frac{q_2^{max}}{S} \\ \frac{-q_1^{max}}{S} & 0 \\ 0 & \frac{-q_2^{max}}{S} \end{bmatrix} \mathrm{u},$$

where the state vector $x \in \mathbb{R}^4$, the input vector $u \in \mathbb{R}^2$, $\tau_i = \frac{S}{a_i}\sqrt{\frac{2h_{\mathrm{e}}^i}{g}}$ denotes the time constant for tank i and $S = 0.02$, $a_i = 5.8 \cdot 10^{-5}$. Using zero-order hold method with a sampling time of $T = 5s$ we obtain the discrete time model of type (10.2):

$$\begin{aligned} x_1(k+1) &= A_{11}x_1(k) + B_{11}u_1(k) + B_{12}u_2(k), \\ x_2(k+1) &= A_{22}x_2(k) + B_{22}u_2(k) + B_{21}u_1(k), \end{aligned}$$

with the partition $x_1 \leftarrow [x_1\ x_4]^T$ and $x_2 \leftarrow [x_2\ x_3]^T$. For the input constraints we consider the practical constraint of the ratios of the three way valves for our plant, i.e $u \in [0.3, 0.8] - \gamma^{\mathrm{e}}$. We also consider state constraints $0 \leq x \leq 0.31 - h^{\mathrm{e}}$. For the stage cost we have taken the weighting matrices to be $Q_i = I_2$ and $R_i = 0.1$. Since the matrices A_{ii} are stable, in order to ensure stability of the MPC scheme we can compute matrices P_i for the final costs as the solutions of the discrete Lyapunov equations [19]: $A_{ii}^T P_i A_{ii} - P_i + Q_i = 0$.

For a prediction horizon $N = 20$ and starting from $h_0 = 0.3$m, the simulated closed-loop trajectories for the levels using Algorithm 10.1 are displayed in Fig. 10.1. Note that the closed-loop system is driven to the equilibrium point h^{e}.

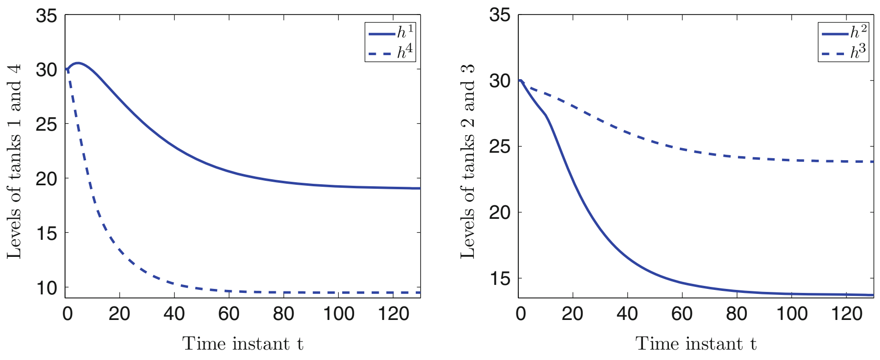

Fig. 10.1 Trajectories of the states (levels) for the four tanks

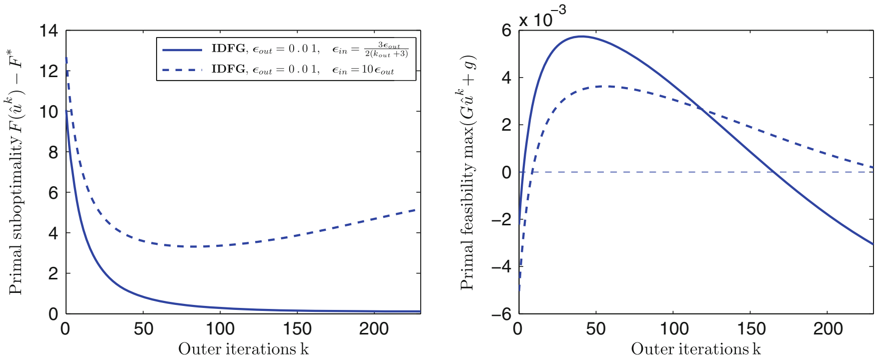

Fig. 10.2 Primal suboptimality and infeasibility for Algorithm 10.1

In Fig. 10.2 we represent the primal suboptimality and infeasibility for the MPC optimization problem solved at step $k = 10$ using Algorithm 10.1. We fix $\epsilon_{\text{out}} = 0.01$. The continuous line corresponds to ϵ_{in} and ϵ_{c} computed according to Sect. 10.3.2, while the dotted line corresponds to $\epsilon_{\text{in}} = 10\epsilon_{\text{out}}$. We can observe that Algorithm 10.1 is sensitive due to error accumulations.

10.6 Conclusions

Motivated by MPC problems for network systems, we have proposed a dual based method for solving constrained MPC problems in network systems. We moved the coupling constraints in the cost using duality theory and introduced a tightened version of the dual MPC problem. We solved the inner subproblems only up to a certain accuracy by means of a parallel coordinate descent method for which we have linear convergence. For solving the outer problems we developed an inexact dual fast gradient scheme. We proved primal feasibility for our original MPC problem and derived upper bounds on dual and primal suboptimality. We also discussed some implementation issues of the new algorithms for distributed MPC and tested them on a practical example.

For future research we intend to apply these newly developed algorithms to other practical problems and implement them in a distributed setting using MPI.

10.7 Appendix

Proof of Theorem 2: Using (10.15) and (10.17) and the convexity of F and h together with the fact that $Q \subseteq \mathbb{R}^m_+$, we obtain:

$$F(\hat{\mathbf{u}}^p) - d_{\epsilon_c}(\mu^p) \le -\max_{\lambda \in Q} -\frac{4L_d}{(p+1)^2}\|\lambda\|^2 + \langle \lambda, h(\hat{\mathbf{u}}^p) + \epsilon_c \mathbf{e}\rangle + \frac{p+3}{3}\epsilon_{in}.$$

Since $\max_{\lambda \in Q} -\frac{4L_d}{(p+1)^2}\|\lambda\|^2 + \langle \lambda, h(\hat{\mathbf{u}}^p) + \epsilon_c \mathbf{e}\rangle \ge 0$ and taking into account that $d_{\epsilon_c}(\mu^p) \le F^*_{\epsilon_c}$, we have: $F(\hat{\mathbf{u}}^p) - F^*_{\epsilon_c} \le \frac{p+3}{3}\epsilon_{in}$. We can write further:

$$\begin{aligned} F^*_{\epsilon_c} &= \max_{\lambda \in Q} d_{\epsilon_c}(\lambda) = \max_{\lambda \in Q}\min_{\mathbf{u}\in\mathbf{U}} F(\mathbf{u}) + \langle \lambda, h(\mathbf{u}) + \epsilon_c \mathbf{e}\rangle \\ &\le \max_{\lambda \in Q}\min_{\mathbf{u}\in\mathbf{U}} F(\mathbf{u}) + \langle \lambda, h(\mathbf{u})\rangle + \max_{\lambda \in Q}\langle \lambda, \epsilon_c \mathbf{e}\rangle = F^* + \sqrt{m}R\epsilon_c, \end{aligned}$$

which combined with the previous inequality proves (10.18). □

Proof of Theorem 3: Recall that $h(\mathbf{u}) = \mathbf{G}\mathbf{u} + \mathbf{g}$. Using (10.15) and convexity of F and h, we have:

$$\max_{\lambda \ge 0} -\frac{4L_d}{(p+1)^2}\|\lambda\|^2 + \langle \lambda, h(\hat{\mathbf{u}}^p) + \epsilon_c \mathbf{e}\rangle \le \frac{p+3}{3}\epsilon_{in} + d_{\epsilon_c}(\mu^p) - F(\hat{\mathbf{u}}^p).$$

For the second term of the right hand-side we have:

$$\begin{aligned} &d_{\epsilon_c}(\mu^p) - F(\hat{\mathbf{u}}^p) \le d_{\epsilon_c}(\lambda^*) - F(\hat{\mathbf{u}}^p) = \min_{\mathbf{u}\in\mathbf{U}} F(\mathbf{u}) + \langle \lambda^*, h(\mathbf{u}) + \epsilon_c \mathbf{e}\rangle - F(\hat{\mathbf{u}}^p) \\ &\le F(\hat{\mathbf{u}}^p) + \langle \lambda^*, h(\hat{\mathbf{u}}^p) + \epsilon_c \mathbf{e}\rangle - F(\hat{\mathbf{u}}^p) = \langle \lambda^*, h(\hat{\mathbf{u}}^p) + \epsilon_c \mathbf{e}\rangle \le \|\lambda^*\|\,\|[h(\hat{\mathbf{u}}^p) + \epsilon_c \mathbf{e}]^+\|, \end{aligned}$$

where in the last inequality we used that $\langle \lambda, y\rangle \le \|\lambda\|\,\|[y]^+\|$ for any $y \in \mathbb{R}^m$ and $\lambda \ge 0$. Using now the fact that:

$$\max_{\lambda \ge 0} -\frac{4L_d}{(p+1)^2}\|\lambda\|^2 + \langle \lambda, h(\hat{\mathbf{u}}^p) + \epsilon_c \mathbf{e}\rangle = \frac{(p+1)^2}{16L_d}\|[h(\hat{\mathbf{u}}^p) + \epsilon_c \mathbf{e}]^+\|^2$$

and introducing the notation $\alpha = h(\hat{\mathbf{u}}^p) + \epsilon_c \mathbf{e}$, we obtain the following second order inequality in $\|[\alpha]^+\|$: $\frac{(p+1)^2}{16L_d}\|[\alpha]^+\|^2 - \|\lambda^*\|\,\|[\alpha]^+\| - \frac{p+3}{3}\epsilon_{in} \le 0$. Therefore, $\|[\alpha]^+\|$ must be less or equal than the largest root of the corresponding second-order equation, from which, together with (10.12), we get the result. □

Acknowledgments The research leading to these results has received funding from: the European Union (FP7/2007–2013) under Grant agreement no 248940; CNCS (project TE-231, 19/11.08.2010); ANCS (project PN II, 80EU/2010); POSDRU/89/1.5/S/62557.

References

1. I. Alvarado, D. Limon, D. Munoz de la Pena, J.M. Maestre, M.A. Ridao, H. Scheu, W. Marquardt, R.R. Negenborn, B. De Schutter, F. Valencia, J. Espinosa, A comparative analysis of distributed MPC techniques applied to the HD-MPC four-tank benchmark. J. Process Control **21**(5), 800–815 (2011)

2. D.P. Bertsekas, J.N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods* (Prentice Hall, Englewood, 1989)
3. E. Camponogara, H.F. Scherer, Distributed optimization for model predictive control of linear dynamic networks with control-input and output constraints. IEEE Trans. Autom. Sci. Eng. **8**(1), 233–242 (2011)
4. O. Devolder, F. Glineur, Y. Nesterov. First-order methods of smooth convex optimization with inexact oracle. Technical report, CORE Discussion Paper 2011/02, UCL, Belgium (2011)
5. M.D. Doan, T. Keviczky, B. De Schutter, A distributed optimization-based approach for hierarchical model predictive control of large-scale systems with coupled dynamics and constraints, in *Proceedings of 50th IEEE Conference on Decision and, Control*, pp. 5236–5241 (2011)
6. M. Farina, R. Scattolini, Distributed predictive control: a non-cooperative algorithm with neighbor-to-neighbor communication for linear systems. Automatica **48**(6), 1088–1096 (2012)
7. K.C. Kiwiel, T. Larsson, P.O. Lindberg, Lagrangian relaxation via ballstep subgradient methods. Math. Oper. Res. **32**(3), 669–686 (2007)
8. Y. Kuwata, A. Richards, T. Schouwenaars, J.P. How, Distribted robust receding horizon control for multivariable guidance. IEEE Trans. Control Syst. Technol. **15**(4), 627–641 (2007)
9. T. Larsson, M. Patriksson, A. Stromberg, Ergodic convergence in subgradient optimization. Optim. Methods Softw. **9**, 93–120 (1998)
10. I. Necoara, D. Clipici. An efficient parallel coordinate descent algorithm for distributed MPC. J. Process Control **23**(3), 243–253 (2013)
11. I. Necoara, V. Nedelcu, I. Dumitrache, Parallel and distributed optimization methods for estimation and control in networks. J. Process Control **21**, 756–766 (2011)
12. I. Necoara, J. Suykens, Application of a smoothing technique to decomposition in convex optimization. IEEE Trans. Autom. Control **53**(11), 2674–2679 (2008)
13. A. Nedic, A. Ozdaglar, Approximate primal solutions and rate analysis for dual subgradient methods. SIAM J. Optim. **19**, 1757–1780 (2009)
14. R.R. Negenborn, B. De Schutter, J. Hellendoorn, Multi-agent model predictive control for transportation networks: serial versus parallel schemes. Eng. Appl. Artif. Intell. **21**(3), 353–366 (2008)
15. Y. Nesterov, Smooth minimization of non-smooth functions. Math. Program. **103**(1), 127–152 (2004)
16. Y. Nesterov, Efficiency of coordinate descent methods on huge-scale optimization problems. Technical report, CORE Discussion Paper 2010/2, UCL, Belgium (2010)
17. A. Richards, J.P. How, Robust distributed model predictive control. Int. J. Control **80**(9), 1517–1531 (2007)
18. S. Richter, M. Morari, C.N. Jones, Towards computational complexity certification for constrained MPC based on lagrange relaxation and the fast gradient method, in *Proceedings of 50th IEEE Conference on Decision and, Control*, pp. 5223–5229 (2011)
19. P.O.M. Scokaert, D.Q. Mayne, J.B. Rawlings, Suboptimal model predictive control (feasibility implies stability). IEEE Trans. Autom. Control **44**(3), 648–654 (1999)
20. B.T. Stewart, A.N. Venkat, J.B. Rawlings, S.J. Wright, G. Pannocchia, Cooperative distributed model predictive control. Syst. Control Lett. **59**, 460–469 (2010)