Generalized Accelerated Gradient Methods for DMPC Based on Dual Decomposition

Pontus Giselsson and Anders Rantzer

Abstract We consider distributed model predictive control (DMPC) where a sparse centralized optimization problem without a terminal cost or a terminal constraint set is solved in distributed fashion. Distribution of the optimization algorithm is enabled by dual decomposition. Gradient methods are usually used to solve the dual problem resulting from dual decomposition. However, gradient methods are known for their slow convergence rate, especially for ill-conditioned problems. This is not desirable in DMPC where the amount of communication should be kept as low as possible. In this chapter, we present a distributed optimization algorithm applied to solve optimization problems arising in DMPC that has significantly better convergence rate than the classical gradient method. This improved convergence rate is achieved by using accelerated gradient methods instead of standard gradient methods and by in a well-defined manner, incorporating Hessian information into the gradient-iterations. We also present a stopping condition to the distributed optimization algorithm that ensures feasibility, stability and closed loop performance of the DMPC-scheme, without using a stabilizing terminal cost or terminal constraint set.

Pontus Giselsson
Department of Automatic Control, Lund University, e-mail: pontusg@control.lth.se

Anders Rantzer
Department of Automatic Control, Lund University, e-mail: rantzer@control.lth.se

1 Introduction

Dual decomposition techniques have often been applied to solve distributed optimization problems with a strongly convex objective arising in distributed model predictive control. Traditionally, the dual problem resulting from dual decomposition is solved using gradient methods. These methods are known to have bad convergence rate properties $O(1/p)$ with p the iteration number [12], especially for ill-conditioned problems. Thus, to implement a DMPC controller based on dual decomposition and a gradient method would require extensive communication. In this chapter we propose a method that reduces the communication requirement significantly when solving the dual problem obtained from dual decomposition. The communication reduction is made possible by two main improvements compared to the classical gradient method. The first improvement is to use accelerated gradient methods, which converges as $O(1/p^2)$, instead of gradient methods to solve the dual problem. For more on accelerated gradient methods, the reader is referred to [12, 1, 14, 15]. Also, the optimal step size for the algorithm is provided which is important for performance reasons. The other improvement is on the gradient step. In every iteration in gradient or accelerated gradient methods, a quadratic upper bound with the same curvature in every direction is minimized to compute the new iterate. If this quadratic upper bound does not well approximate the cost function, the number of iterations can be significant. By allowing for different curvature in different directions of the quadratic upper bound, a much closer fit between the cost function and the upper bound can be obtained, especially for ill-conditioned problems. This can reduce significantly the number of iterations, hence the amount of communication, needed to achieve the desired accuracy of the solution. The distributed optimization algorithm with improved convergence rate was published in [3].

We also present a stopping condition for the presented distributed optimization algorithm. The stopping condition is developed to guarantee feasibility, stability, and a prespecified performance of the closed loop system. Traditionally, a terminal cost and a terminal constraint set is used to prove stability in MPC. These are usually dense, i.e., involve all state variables. In dual decomposition, the cost function should be separable and the constraints should be sparse. Hence, we cannot rely on a terminal cost or a terminal constraint set prove stability in DMPC based on dual decomposition. However, stability and performance of the closed loop system can be established by choosing the control horizon such that the optimal value function is decreasing with a certain amount in each time step. Further, feasibility in duality-based optimization can only be guaranteed in the limit. To guarantee feasibility with a finite number of iterations, an adaptive constraint tightening approach is presented. The constraint tightening enables a feasible solution within finite number of iterations and the adaptation adapts the amount of constraint tightening to guarantee that the optimal value function is decreasing in each time step. This implies that feasibility, stability, and a prespecified performance of the closed loop system are guaranteed using the presented stopping condition. The stopping condition was published in [8].

2 Problem Formulation

We consider control of linear dynamical systems of the form

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k), \qquad \mathbf{x}(0) = \bar{\mathbf{x}}$$

where $\mathbf{x} \in \mathbb{R}^{n_x}$, $\mathbf{u} \in \mathbb{R}^{n_u}$, $\mathbf{A} \in \mathbb{R}^{n_x \times n_x}$, and $\mathbf{B} \in \mathbb{R}^{n_x \times n_u}$. We assume that the pair $(\mathbf{A}, \mathbf{B})$ is controllable. The state and control vectors are partitioned according to

$$\mathbf{x} = [\mathbf{x}_1^T, \ldots, \mathbf{x}_{|\mathcal{N}|}^T]^T, \qquad \mathbf{u} = [\mathbf{u}_1^T, \ldots, \mathbf{u}_{|\mathcal{N}|}^T]^T$$

where $\mathbf{x}_i \in \mathbb{R}^{n_{x_i}}$, $\mathbf{u}_i \in \mathbb{R}^{n_{u_i}}$, for all $i \in \mathcal{N}$ are referred to as local variables, $\mathcal{N} = \{1, \ldots, |\mathcal{N}|\}$ is the set of subsystems, and $|\mathcal{N}|$ is the number of subsystems. The dynamics matrices are partitioned according to the state and control variable partitions

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \cdots & \mathbf{A}_{1|\mathcal{N}|} \\ \vdots & \ddots & \vdots \\ \mathbf{A}_{|\mathcal{N}|1} & \cdots & \mathbf{A}_{|\mathcal{N}||\mathcal{N}|} \end{pmatrix}, \qquad \mathbf{B} = \begin{pmatrix} \mathbf{B}_{11} & \cdots & \mathbf{B}_{1|\mathcal{N}|} \\ \vdots & \ddots & \vdots \\ \mathbf{B}_{|\mathcal{N}|1} & \cdots & \mathbf{B}_{|\mathcal{N}||\mathcal{N}|} \end{pmatrix}$$

where $\mathbf{A}_{ij} = \mathbb{R}^{n_{x_i} \times n_{x_j}}$ and $\mathbf{B}_{ij} = \mathbb{R}^{n_{x_i} \times n_{u_j}}$. We assume that the matrices have a sparse structure, i.e., that some $\mathbf{A}_{ij} = 0$ and $\mathbf{B}_{ij} = 0$. The neighboring interaction is defined by the following sets

$$\mathcal{N}_i = \{j \in \mathcal{N} \mid \mathbf{A}_{ij} \neq 0 \text{ or } \mathbf{B}_{ij} \neq 0\},$$
$$\mathcal{M}_i = \{j \in \mathcal{N} \mid \mathbf{A}_{ji} \neq 0 \text{ or } \mathbf{B}_{ji} \neq 0\}.$$

Using the introduced local variables and the neighborhood sets, the local dynamics is described by

$$\mathbf{x}_i(k+1) = \sum_{j \in \mathcal{N}_i} \left(\mathbf{A}_{ij}\mathbf{x}_j(k) + \mathbf{B}_{ij}\mathbf{u}_j(k)\right), \qquad \mathbf{x}_i(0) = \bar{\mathbf{x}}_i.$$

There are constraints on the local state and control variables, they should satisfy $\mathbf{x}_i \in \mathcal{X}_i$, and $\mathbf{u}_i \in \mathcal{U}_i$ where

$$\mathcal{X}_i = \{\mathbf{x}_i \in \mathbb{R}^{n_{x_i}} \mid \mathbf{C}_{x,i}\mathbf{x}_i \leq \mathbf{d}_{x,i}\}, \quad \mathcal{U}_i = \{\mathbf{u}_i \in \mathbb{R}^{n_{u_i}} \mid \mathbf{C}_{u,i}\mathbf{u}_i \leq \mathbf{d}_{u,i}\}$$

and $\mathbf{C}_{x,i}$, $\mathbf{C}_{u,i}$, $\mathbf{d}_{x,i} > 0$, and $\mathbf{d}_{u,i} > 0$ are real matrices/vectors with appropriate dimensions. The assumption that $\mathbf{d}_{x,i} > 0$ and $\mathbf{d}_{u,i} > 0$ implies that $0 \in \operatorname{int}(\mathcal{X}_i)$ and $0 \in \operatorname{int}(\mathcal{U}_i)$. The global constraint sets, $\mathcal{X}$ and $\mathcal{U}$, are products of local sets, i.e.,

$$\mathcal{X} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_{|\mathcal{N}|}, \qquad \mathcal{U} = \mathcal{U}_1 \times \ldots \times \mathcal{U}_{|\mathcal{N}|}.$$

The total number of inequalities describing the sets $\mathcal{X}$ and $\mathcal{U}$ is denoted by n_c. We assume quadratic local stage-cost functions, i.e., local stage-cost functions of the form

$$\ell_i(\mathbf{x}_i, \mathbf{u}_i) = \frac{1}{2}\left(\mathbf{x}_i^T \mathbf{Q}_i \mathbf{x}_i + \mathbf{u}_i^T \mathbf{R}_i \mathbf{u}_i\right)$$

where cost matrices $\mathbf{Q}_i \in \mathbb{R}^{n_{x_i} \times n_{x_i}}$ and $\mathbf{R}_i \in \mathbb{R}^{n_{u_i} \times n_{u_i}}$ for all $i \in \mathcal{N}$ are assumed symmetric and positive definite. This gives the following stage-cost for the full system

$$\ell(\mathbf{x}, \mathbf{u}) = \sum_{i=1}^{|\mathcal{N}|} \ell_i(\mathbf{x}_i, \mathbf{u}_i) = \frac{1}{2}\sum_{i=1}^{|\mathcal{N}|}\left(\mathbf{x}_i^T \mathbf{Q}_i \mathbf{x}_i + \mathbf{u}_i^T \mathbf{R}_i \mathbf{u}_i\right) = \frac{1}{2}\left(\mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{u}^T \mathbf{R} \mathbf{u}\right)$$

where $\mathbf{Q} = \mathrm{blkdiag}(\mathbf{Q}_1, \ldots, \mathbf{Q}_{|\mathcal{N}|})$ and $\mathbf{R} = \mathrm{blkdiag}(\mathbf{R}_1, \ldots, \mathbf{R}_{|\mathcal{N}|})$. For future reference we also introduce

$$\ell^*(\mathbf{x}) := \min_{\mathbf{u} \in \mathcal{U}} \ell(\mathbf{x}, \mathbf{u}) = \ell(\mathbf{x}, \mathbf{0}) = \frac{1}{2}\mathbf{x}^T \mathbf{Q} \mathbf{x}.$$

We use prediction horizon $N_p = N$ and control horizon $N_c = N$. Also, neither a terminal cost nor a terminal constraint set is used in the DMPC optimization problem formulation. Hence, the optimization problem to be solved for initial condition $\bar{\mathbf{x}}$ in the DMPC scheme is:

$$\begin{aligned} V_N(\bar{\mathbf{x}}) := \min_{\hat{\mathbf{x}}, \hat{\mathbf{u}}} \; & \sum_{l=0}^{N-1} \frac{1}{2}(\hat{\mathbf{x}}^T(l)\mathbf{Q}\hat{\mathbf{x}}(l) + \hat{\mathbf{u}}^T(l)\mathbf{R}\hat{\mathbf{u}}(l)) \\ \text{s.t. } & (\hat{\mathbf{x}}(l), \hat{\mathbf{u}}(l)) \in \mathcal{X} \times \mathcal{U}, \qquad l = 0, \ldots, N-1 \\ & \hat{\mathbf{x}}(l+1) = A\hat{\mathbf{x}}(l) + B\hat{\mathbf{u}}(l), \quad l = 0, \ldots, N-2 \\ & \hat{\mathbf{x}}(0) = \bar{\mathbf{x}} \end{aligned} \tag{1}$$

where $\hat{\mathbf{x}}(l)$ and $\hat{\mathbf{u}}(l)$ denote the predicted state and control variables l steps ahead. To describe the optimization problem in a more compact form, the following stacked vectors are introduced

$$\mathbf{z}_i = [\hat{\mathbf{x}}_i^T(0), \ldots, \hat{\mathbf{x}}_i^T(N-1), \hat{\mathbf{u}}_i^T(0), \ldots, \hat{\mathbf{u}}_i^T(N-1)]^T$$

for all $i \in \mathcal{N}$ and $\mathbf{z} = [\mathbf{z}_1^T, \ldots, \mathbf{z}_{|\mathcal{N}|}^T]^T$. This implies that the optimization problem (1) more compactly can be written as

$$\begin{aligned} V_N(\bar{\mathbf{x}}) := \min_{\mathbf{z}} \; & \tfrac{1}{2}\mathbf{z}^T \mathbf{H} \mathbf{z} \\ \text{s.t. } & \mathbf{F}\mathbf{z} = \mathbf{g}\bar{\mathbf{x}} \\ & \mathbf{C}\mathbf{z} \leq \mathbf{d} \end{aligned} \tag{2}$$

where

$$\begin{aligned}
\mathbf{H} &= \text{blkdiag}(\mathbf{H}_1, \ldots, \mathbf{H}_{|\mathcal{N}|}), & \bar{\mathbf{x}} &= [\bar{\mathbf{x}}_1^T, \ldots, \bar{\mathbf{x}}_{|\mathcal{N}|}^T]^T, \\
\mathbf{F} &= [\mathbf{F}_1^T, \ldots, \mathbf{F}_{|\mathcal{N}|}^T]^T, & \mathbf{g} &= [\mathbf{g}_1^T, \ldots, \mathbf{g}_{|\mathcal{N}|}^T]^T, \\
\mathbf{C} &= \text{blkdiag}(\mathbf{C}_1, \ldots, \mathbf{C}_{|\mathcal{N}|}), & \mathbf{d} &= [\mathbf{d}_1^T, \ldots, \mathbf{d}_{|\mathcal{N}|}^T]^T
\end{aligned}$$

and

$$\begin{aligned}
\mathbf{H}_i &= \text{blkdiag}(\mathbf{Q}_i, \ldots, \mathbf{Q}_i, \mathbf{R}_i, \ldots, \mathbf{R}_i), \\
\mathbf{F}_i &= [\mathbf{F}_{i1}, \ldots, \mathbf{F}_{i|\mathcal{N}|}], \\
\mathbf{F}_{ij} &= \begin{cases}
\begin{bmatrix}
\mathbf{0} & & \mathbf{B}_{ij} & & \\
\mathbf{A}_{ij} & \ddots & & \ddots & \\
& \ddots & \ddots & & \ddots \\
& & \mathbf{A}_{ij} & \mathbf{0} & & \mathbf{B}_{ij}
\end{bmatrix}, & j \in \mathcal{N}_i \backslash \{i\} \\
\begin{bmatrix}
-\mathbf{I} & & \mathbf{B}_{ii} & & \\
\mathbf{A}_{ii} & \ddots & & \ddots & \\
& \ddots & \ddots & & \ddots \\
& & \mathbf{A}_{ii} & -\mathbf{I} & & \mathbf{B}_{ii}
\end{bmatrix}, & j = i \\
\mathbf{0}, & & j \notin \mathcal{N}_i
\end{cases} \\
\mathbf{g}_i &= [\mathbf{g}_{i1}, \ldots, \mathbf{g}_{i|\mathcal{N}|}], \\
\mathbf{g}_{ij} &= \begin{cases} [-\mathbf{A}_{ij}^T, \mathbf{0}^T, \ldots, \mathbf{0}^T]^T, & j \in \mathcal{N}_i \\ \mathbf{0}, & j \notin \mathcal{N}_i \end{cases} \\
\mathbf{C}_i &= \text{blkdiag}(\mathbf{C}_{x,i}, \ldots, \mathbf{C}_{x,i}, \mathbf{C}_{u,i}, \ldots, \mathbf{C}_{u,i}), \\
\mathbf{d}_i &= [\mathbf{d}_{x,i}^T, \ldots, \mathbf{d}_{x,i}^T, \mathbf{d}_{u,i}^T, \ldots, \mathbf{d}_{u,i}^T]^T.
\end{aligned}$$

The optimization problem (2) is solved in every time instant in the DMPC controller with the latest measurement as initial condition to the state predictions. Communication between subsystems i and j is allowed if $j \in \mathcal{N}_i \cup \mathcal{M}_i$.

3 Description of the DMPC Method

In this section the proposed DMPC methodology is presented. We present a distributed algorithm based dual decomposition and a generalized accelerated gradient method. We also present a stopping condition that can be used to guarantee feasibility, stability, and a prespecified performance of the closed loop system.

3.1 Dual Problem Formulation

We introduce dual variables $\boldsymbol{\lambda} \in \mathbb{R}^{n_\lambda}$ for the equality constraints and dual variables $\boldsymbol{\mu} \in \mathbb{R}^{n_\mu}_{\geq 0}$ for the inequality constraints in (2), where $n_\lambda = (N-1)(n_x + n_u)$ and $n_\mu = Nn_c$. This gives the following dual problem

$$\max_{\boldsymbol{\mu}\geq 0, \boldsymbol{\lambda}} \min_{\mathbf{z}} \frac{1}{2}\mathbf{z}^T\mathbf{H}\mathbf{z} + \boldsymbol{\lambda}^T(\mathbf{F}\mathbf{z} - \mathbf{g}\bar{\mathbf{x}}) + \boldsymbol{\mu}^T(\mathbf{C}\mathbf{z} - \mathbf{d}).$$

By solving the inner minimization problem explicitly, the dual problem becomes

$$\max_{\boldsymbol{\mu}\geq 0, \boldsymbol{\lambda}} -\frac{1}{2}(\mathbf{F}^T\boldsymbol{\lambda} + \mathbf{C}^T\boldsymbol{\mu})^T\mathbf{H}^{-1}(\mathbf{F}^T\boldsymbol{\lambda} + \mathbf{C}^T\boldsymbol{\mu}) - \boldsymbol{\lambda}^T\mathbf{g}\bar{\mathbf{x}} - \boldsymbol{\mu}^T\mathbf{d}. \tag{3}$$

The dual function for initial condition $\bar{\mathbf{x}}$ is defined as

$$D_N(\bar{\mathbf{x}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) := -\frac{1}{2}(\mathbf{F}^T\boldsymbol{\lambda} + \mathbf{C}^T\boldsymbol{\mu})^T\mathbf{H}^{-1}(\mathbf{F}^T\boldsymbol{\lambda} + \mathbf{C}^T\boldsymbol{\mu}) - \boldsymbol{\lambda}^T\mathbf{g}\bar{\mathbf{x}} - \boldsymbol{\mu}^T\mathbf{d} \tag{4}$$

which is quadratic, concave, and differentiable with gradient

$$\nabla D_N(\bar{\mathbf{x}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = -\begin{bmatrix}\mathbf{F}\\ \mathbf{C}\end{bmatrix}\mathbf{H}^{-1}(\mathbf{F}^T\boldsymbol{\lambda} + \mathbf{C}^T\boldsymbol{\mu}) - \begin{bmatrix}\mathbf{g}\bar{\mathbf{x}}\\ \mathbf{d}\end{bmatrix}. \tag{5}$$

We partition the dual variables into local dual variables $\boldsymbol{\lambda}_i \in \mathbb{R}^{n_{\lambda_i}}$ and $\boldsymbol{\mu}_i \in \mathbb{R}^{n_{\mu_i}}_{\geq 0}$ according to $\boldsymbol{\lambda} = [\boldsymbol{\lambda}_1^T, \ldots, \boldsymbol{\lambda}_{|\mathcal{N}|}^T]^T$ and $\boldsymbol{\mu} = [\boldsymbol{\mu}_1^T, \ldots, \boldsymbol{\mu}_{|\mathcal{N}|}^T]^T$, where the partitions are introduced according to matrices $\mathbf{F}$ and $\mathbf{C}$ respectively. This gives that the dual function gradients w.r.t. to local dual variables are given by

$$\begin{aligned}
\nabla_{\boldsymbol{\lambda}_i} D_N(\bar{\mathbf{x}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) &= \sum_{j\in\mathcal{N}_i}\Big[-\mathbf{F}_{ij}\mathbf{H}_j^{-1}\Big(\sum_{l\in\mathcal{M}_j}\mathbf{F}_{lj}^T\boldsymbol{\lambda}_l + \mathbf{C}_j^T\boldsymbol{\mu}_j\Big) - \mathbf{g}_{ij}\bar{\mathbf{x}}_j\Big]\\
\nabla_{\boldsymbol{\mu}_i} D_N(\bar{\mathbf{x}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) &= -\mathbf{C}_i\mathbf{H}_i^{-1}\Big(\sum_{j\in\mathcal{M}_i}\mathbf{F}_{ji}^T\boldsymbol{\lambda}_j + \mathbf{C}_i^T\boldsymbol{\mu}_i\Big) - \mathbf{d}_i.
\end{aligned}$$

By setting the primal variable

$$\mathbf{z}_i = -\mathbf{H}_i^{-1}\Big(\sum_{j\in\mathcal{M}_i}\mathbf{F}_{ji}^T\boldsymbol{\lambda}_j + \mathbf{C}_i^T\boldsymbol{\mu}_i\Big) \tag{6}$$

we get that the local gradients are described by

$$\nabla_{\boldsymbol{\lambda}_i} D_N(\bar{\mathbf{x}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \sum_{j\in\mathcal{N}_i}\left(\mathbf{F}_{ij}\mathbf{z}_j - \mathbf{g}_{ij}\bar{\mathbf{x}}_j\right), \quad \nabla_{\boldsymbol{\mu}_i} D_N(\bar{\mathbf{x}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \mathbf{C}_i\mathbf{z}_i - \mathbf{d}_i. \tag{7}$$

Further, the gradient of the dual function ∇D_N is Lipschitz continuous with constant

$$L = \|[\mathbf{F}^T \ \mathbf{C}^T]^T \mathbf{H}^{-1} [\mathbf{F}^T \ \mathbf{C}^T]\|. \tag{8}$$

3.2 Gradient-Based Methods

Optimization problems with a differentiable and convex cost function $f : \mathbb{R}^{n_v} \to \mathbb{R}$ and closed and convex constraint set $\mathcal{V}$, can be solved using projected gradient methods. If the gradient to f is Lipschitz continuous with constant L, then the optimal step size for the gradient-step can be shown to be $1/L$. The projected gradient method is described by the following iteration

$$\mathbf{v}^{p+1} = \arg\min_{\mathbf{v}\in\mathcal{V}} \left(\left\| \mathbf{v} - \mathbf{v}^p + \frac{1}{L}\nabla f(\mathbf{v}^p) \right\| \right)$$

where p is the iteration number. The iteration is a step in the gradient direction with step length $1/L$. The resulting point is projected using Euclidean projection onto the feasible set $\mathcal{V}$. By introducing the notation $\langle x, y\rangle = x^T y$, we get the following equivalent formulation of the projected gradient algorithm

$$\mathbf{v}^{p+1} = \arg\min_{\mathbf{v}\in\mathcal{V}} \left[f(\mathbf{v}^p) + \langle \nabla f(\mathbf{v}^p), \mathbf{v} - \mathbf{v}^p \rangle + \frac{L}{2}\|\mathbf{v} - \mathbf{v}^p\|^2 \right].$$

Hence, in a gradient method, a quadratic function with the same curvature in every direction is minimized in every iteration of the algorithm. The quadratic function is actually an upper bound to f since the Lipschitz continuity assumption implies that for every $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^{n_v}$ the following holds

$$f(\mathbf{v}_1) \leq f(\mathbf{v}_2) + \langle \nabla f(\mathbf{v}_2), \mathbf{v}_1 - \mathbf{v}_2 \rangle + \frac{L}{2}\|\mathbf{v}_1 - \mathbf{v}_2\|^2. \tag{9}$$

The classical gradient method is known to have bad convergence rate properties, $O(1/p)$. However, this convergence rate can be improved significantly by instead using accelerated gradient methods that have a convergence rate of $O(1/p^2)$. A simple accelerated projected gradient algorithm is given by the following iterations

$$\begin{aligned} \bar{\mathbf{v}}^p &= \mathbf{v}^p + \frac{p-1}{p+2}(\mathbf{v}^p - \mathbf{v}^{p-1}) \\ \mathbf{v}^{p+1} &= \arg\min_{\mathbf{v}\in\mathcal{V}} \left[f(\bar{\mathbf{v}}^p) + \langle \nabla f(\bar{\mathbf{v}}^p), \mathbf{v} - \bar{\mathbf{v}}^p \rangle + \frac{L}{2}\|\mathbf{v} - \bar{\mathbf{v}}^p\|^2 \right]. \end{aligned}$$

The increase in algorithm complexity compared to the classical gradient method is minor, but the improvement in convergence rate is vast. However, this improvement in convergence rate is not always enough to achieve satisfactory accuracy within a small number of iterations. Another improvement to the convergence rate of the

algorithm can be obtained by letting the quadratic upper bound that is minimized in every iteration, have different curvatures in different directions. The resulting generalized accelerated gradient algorithm is described by the following iterations

$$\begin{aligned}\bar{\mathbf{v}}^p &= \mathbf{v}^p + \frac{p-1}{p+2}(\mathbf{v}^p - \mathbf{v}^{p-1})\\ \mathbf{v}^{p+1} &= \arg\min_{\mathbf{v}\in\mathcal{V}} \left[f(\bar{\mathbf{v}}^p) + \langle \nabla f(\bar{\mathbf{v}}^p), \mathbf{v} - \bar{\mathbf{v}}^p\rangle + \frac{1}{2}\|\mathbf{v} - \bar{\mathbf{v}}^p\|_{\mathbf{L}}^2 \right]\end{aligned}$$

where $\mathbf{L} \in \mathbb{R}^{n_v \times n_v}$ is a symmetric positive definite matrix that must be chosen such that for every $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^{n_v}$ the following holds

$$f(\mathbf{v}_1) \leq f(\mathbf{v}_2) + \langle \nabla f(\mathbf{v}_2), \mathbf{v}_1 - \mathbf{v}_2\rangle + \frac{1}{2}\|\mathbf{v}_1 - \mathbf{v}_2\|_{\mathbf{L}}^2. \tag{10}$$

This requirement is very similar to the requirement for Lipschitz continuity (9). The only difference is in the quadratic term. The generalized accelerated gradient method can equivalently be written as

$$\bar{\mathbf{v}}^p = \mathbf{v}^p + \frac{p-1}{p+2}(\mathbf{v}^p - \mathbf{v}^{p-1}) \tag{11}$$

$$\mathbf{v}^{p+1} = \arg\min_{\mathbf{v}\in\mathcal{V}} \left(\left\| \mathbf{v} - \bar{\mathbf{v}}^p + \mathbf{L}^{-1}\nabla f(\bar{\mathbf{v}}^p) \right\|_{\mathbf{L}} \right). \tag{12}$$

where $\mathbf{L}^{-1}$ serves as a step matrix for the gradient-step. By choosing the $\mathbf{L}$-matrix wisely, Hessian-information can be introduced to the gradient algorithm, which further improves the convergence rate. The generalized accelerated gradient method differs from the accelerated gradient method only in the weight used in the quadratic penalty. It also shares the same theoretical convergence rate $O(1/p^2)$.

3.3 The Distributed Algorithm

By choosing the $\mathbf{L}$-matrix appropriately, the generalized accelerated gradient method can be applied to solve the dual problem (3). The $\mathbf{L}$-matrix must satisfy

$$\mathbf{L} \succeq [\mathbf{F}^T\ \mathbf{C}^T]^T \mathbf{H}^{-1} [\mathbf{F}^T\ \mathbf{C}^T]$$

which implies that (10) holds for $f = -D_N$ where D_N is the dual function defined in (4). The objective of the algorithm is to enable for a distributed implementation, hence the $\mathbf{L}$-matrix should be chosen to accompany this requirement. We introduce a structural constraint on the $\mathbf{L}$-matrix which is described by the following set

$$\mathcal{L} = \{\mathbf{L}\ \in \mathbb{R}^{(n_\lambda + n_\mu)\times(n_\lambda+n_\mu)} \mid \mathbf{L} = \mathrm{blkdiag}(\mathbf{L}_{\boldsymbol{\lambda}_1}, \ldots, \mathbf{L}_{\boldsymbol{\lambda}_M}, \mathbf{L}_{\boldsymbol{\mu}_1}, \ldots, \mathbf{L}_{\boldsymbol{\mu}_M})\}$$

where $\mathbf{L}_{\lambda_i} \in \mathbb{R}^{n_{\lambda_i} \times n_{\lambda_i}}$ and $\mathbf{L}_{\mu_i} \in \mathbb{R}^{n_{\mu_i} \times n_{\mu_i}}$ for all $i \in \mathcal{N}$. We also introduce $\mathbf{L}_\lambda = \mathrm{blkdiag}(\mathbf{L}_{\lambda_1}, \ldots, \mathbf{L}_{\lambda_M})$ and $\mathbf{L}_\mu = \mathrm{blkdiag}(\mathbf{L}_{\mu_1}, \ldots, \mathbf{L}_{\mu_M})$ for future reference. An $\mathbf{L}$-matrix that satisfies the above mentioned requests can be computed by solving the following semidefinite program

$$\begin{aligned} &\min_{\mathbf{L} \in \mathcal{L}} \ \mathrm{trace}(\mathbf{L}) \\ &\text{s.t. } \mathbf{L} \succeq [\mathbf{F}^T \ \mathbf{C}^T]^T \mathbf{H}^{-1} [\mathbf{F}^T \ \mathbf{C}^T] \\ &\quad \mathbf{L} \succ 0 \end{aligned} \tag{13}$$

The idea behind solving a convex optimization problem to compute the step matrix is similar to the idea used in explicit MPC; by increasing the amount of offline computational burden, the online execution time or amount of communication can be reduced significantly. Using the algorithm description in (11)-(12) and the gradient computations in (5) it is straightforward to verify that the generalized accelerated gradient algorithm when applied to the dual problem becomes

$$\begin{aligned} \mathbf{z}^p &= -\mathbf{H}^{-1}(\mathbf{F}^T \boldsymbol{\lambda}^p + \mathbf{C}^T \boldsymbol{\mu}^p) \\ \bar{\mathbf{z}}^p &= \mathbf{z}^p + \frac{p-1}{p+2}(\mathbf{z}^p - \mathbf{z}^{p-1}) \\ \bar{\boldsymbol{\lambda}}^p &= \boldsymbol{\lambda}^p + \frac{p-1}{p+2}(\boldsymbol{\lambda}^p - \boldsymbol{\lambda}^{p-1}) \\ \boldsymbol{\lambda}^{p+1} &= \bar{\boldsymbol{\lambda}}^p + \mathbf{L}_\lambda^{-1}(\mathbf{F}\bar{\mathbf{z}}^p - \mathbf{g}\bar{x}) \\ \bar{\boldsymbol{\mu}}^p &= \boldsymbol{\mu}^p + \frac{p-1}{p+2}(\boldsymbol{\mu}^p - \boldsymbol{\mu}^{p-1}) \\ \boldsymbol{\mu}^{p+1} &= \arg\min_{\mu \geq 0} \left(\left\| \mu - \bar{\boldsymbol{\mu}}^p - \mathbf{L}_\mu^{-1}(\mathbf{C}\bar{\mathbf{z}}^p + \mathbf{d}) \right\|^2_{\mathbf{L}_\mu} \right) \end{aligned}$$

These computations can be distributed by using the local gradients defined in (6) and (7). The resulting distributed algorithm is presented below.

Algorithm 1 ***Distributed optimization algorithm***

Initialize $\boldsymbol{\lambda}_i^0 = \boldsymbol{\lambda}_i^{-1}, \boldsymbol{\mu}_i^0 = \boldsymbol{\mu}_i^{-1}$ *and* $\mathbf{z}_i^0 = \mathbf{z}_i^{-1}$
In every node, i, the following computations are performed
For $p \geq 0$

1. Update primal variables according to:

$$\begin{aligned} \mathbf{z}_i^p &= -\mathbf{H}_i^{-1}\left(\left(\sum_{j \in \mathcal{M}_i} \mathbf{F}_{ji}^T \boldsymbol{\lambda}_j^p \right) + \mathbf{C}_i^T \boldsymbol{\mu}_i^p \right) \\ \bar{\mathbf{z}}_i^p &= \mathbf{z}_i^p + \frac{p-1}{p+2}(\mathbf{z}_i^p - \mathbf{z}_i^{p-1}) \end{aligned}$$

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

3. Update dual variables according to:

$$\begin{aligned}
\bar{\boldsymbol{\lambda}}_i^p &= \boldsymbol{\lambda}_i^p + \frac{p-1}{p+2}(\boldsymbol{\lambda}_i^p - \boldsymbol{\lambda}_i^{p-1}) \\
\boldsymbol{\lambda}_i^{p+1} &= \bar{\boldsymbol{\lambda}}_i^p + \mathbf{L}_{\lambda_i}^{-1}\Big(\sum_{j\in\mathcal{N}_i}(\mathbf{F}_{ij}\bar{\mathbf{z}}_j^p - \mathbf{g}_{ij}\bar{x}_j)\Big) \\
\bar{\boldsymbol{\mu}}_i^p &= \boldsymbol{\mu}_i^p + \frac{p-1}{p+2}(\boldsymbol{\mu}_i^p - \boldsymbol{\mu}_i^{p-1}) \\
\boldsymbol{\mu}_i^{p+1} &= \arg\min_{\mu\geq 0}\Big(\big\|\mu - \bar{\boldsymbol{\mu}}_i^p - \mathbf{L}_{\mu_i}^{-1}(\mathbf{C}_i\bar{\mathbf{z}}_i^p - \mathbf{d}_i)\big\|_{L_\mu^i}\Big)
\end{aligned}$$

4. Send $\boldsymbol{\lambda}_i^{p+1}$ to each $j \in \mathcal{N}_i$, receive $\boldsymbol{\lambda}_j^{p+1}$ from each $j \in \mathcal{M}_i$

If all $\mathbf{L}_{\mu_i}$, $i \in \mathcal{N}$ are chosen diagonal, the minimization to find $\boldsymbol{\mu}_i^{p+1}$ in Algorithm 1 can be replaced by $\max(0,\cdot)$ which is computed uncostly. Using non-diagonal $\mathbf{L}_{\mu_i}$ gives more elaborate iterations, but the number of iterations to achieve satisfactory accuracy of the solution may decrease significantly. This is advantageous in DMPC, where the amount of communication should be kept as small as possible, without compromising the global closed loop performance.

The algorithm converges in both primal variables $\mathbf{z}_i^p$ and dual function value at the rate $O(1/p^2)$. In [3] methods to compute iteration complexity bounds to achieve a prespecified accuracy of the solution are presented. In [7] similar methods are presented for the case where $\mathbf{L}$ is a multiple of the identity matrix. Also, in [7] it was shown how to precondition the matrices describing the inequality constraints and equality constraints optimally. The optimal preconditioning refers to the preconditioning that minimizes the iteration complexity bound that guarantees a dual function value accuracy.

3.4 Stopping Condition

One drawback of using duality-based optimization in DMPC is that feasibility of the primal problem can be guaranteed only in the limit of iterations. Also, the optimization problem used in Algorithm 1 has neither a terminal cost nor a terminal constraint set. These are usually required to prove stability in MPC. In this section we will briefly present a stopping condition for the duality-based optimization algorithm in Algorithm 1. The stopping condition guarantees feasibility, stability and performance of the closed loop system and it reduces the amount of communication needed since it enables for early termination of the optimization algorithm. The stopping condition is based on relaxed dynamic programming for stability and performance, and adaptive constraint tightening for feasibility. We start by describing relaxed dynamic programming when applied to MPC with optimization problems without a terminal cost or a terminal constraint set.

Relaxed dynamic programming applied to MPC states that if the control horizon N is such that for every $\mathbf{x} \in \mathcal{X}$ the following holds

$$V_N(\mathbf{x}) \geq V_N(\mathbf{A}\mathbf{x} + \mathbf{B}\nu_N(\mathbf{x})) + \alpha\ell(\mathbf{x}, \nu_N(\mathbf{x})) \tag{14}$$

where $\alpha \in (0, 1)$, V_N is the optimal value function to the optimization problem (2) without terminal constraint set of terminal cost, and ν_N is the MPC feedback control law obtained by solving (2) and applying the first control action in optimal control trajectory in every sample. Then we get asymptotic stability and closed loop performance as specified by

$$\alpha \sum_{k=0}^{\infty} \ell(\mathbf{x}(k), \nu_N(\mathbf{x}(k))) \leq V_\infty(\mathbf{x}(0)) \tag{15}$$

where $\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\nu_N(\mathbf{x}(k))$. A method to find control horizon N such that (14) holds is presented in [10]. This was further extended in [9] where it was shown how to compute the minimal control horizon for systems that satisfies a certain controllability assumption on the stage costs. Another, more explicit, characterization of the relation between the control horizon N and the performance parameter α was presented in [8]. Once a control horizon N is known such that (14) holds for every $\mathbf{x} \in \mathcal{X}$, performance and stability is guaranteed by computing the optimal solution to (2). However, stability and performance can be guaranteed also for suboptimal solutions, which indicates a potential benefit of using a stopping condition that ensures this.

The issue that feasibility can be guaranteed only in the limit of iterations in duality-based optimization, is addressed by an adaptive constraint tightening approach. We use the following optimization problem with tightened constraint sets

$$\begin{aligned} V_N^\delta(\bar{\mathbf{x}}) := \min_{\mathbf{z}} \ & \tfrac{1}{2}\mathbf{z}^T\mathbf{H}\mathbf{z} \\ \text{s.t. } & \mathbf{F}\mathbf{z} = \mathbf{g}\bar{\mathbf{x}} \\ & \mathbf{C}\mathbf{z} \leq (1-\delta)\mathbf{d} \end{aligned} \tag{16}$$

where $\delta \in (0, 1)$ specifies the relative constraint tightening used and the matrices in (16) are specified at the end of Section 2. The objective is to choose constraint tightening δ such that it can be guaranteed that (14) holds and that feasibility can be guaranteed with finite number of iterations. The problem (16) is solved through the dual problem

$$\max_{\boldsymbol{\lambda},\boldsymbol{\mu}\geq 0} -\frac{1}{2}(\mathbf{F}^T\boldsymbol{\lambda} + \mathbf{C}^T\boldsymbol{\mu})^T\mathbf{H}^{-1}(\mathbf{F}^T\boldsymbol{\lambda} + \mathbf{C}^T\boldsymbol{\mu}) - \boldsymbol{\lambda}^T\mathbf{g}\bar{\mathbf{x}} - \boldsymbol{\mu}^T\mathbf{d}(1-\delta)$$

using Algorithm 1. The dual function for initial condition $\bar{\mathbf{x}}$ and with constraint tightening δ is defined as

$$D_N^\delta(\bar{\mathbf{x}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) := -\frac{1}{2}(\mathbf{F}^T\boldsymbol{\lambda} + \mathbf{C}^T\boldsymbol{\mu})^T\mathbf{H}^{-1}(\mathbf{F}^T\boldsymbol{\lambda} + \mathbf{C}^T\boldsymbol{\mu}) - \boldsymbol{\lambda}^T\mathbf{g}\bar{\mathbf{x}} - \boldsymbol{\mu}^T\mathbf{d}(1-\delta)$$

To online guarantee that the condition (14) holds, a lower bound to the l.h.s. is needed and an upper bound to the r.h.s. is needed. A lower bound to the l.h.s. of (14) is obtained by using the following lemma which is proven in [8, Lemma 1].

Lemma 1 *For every $\bar{\mathbf{x}} \in \mathbb{R}^{n_x}$, $\boldsymbol{\lambda} \in \mathbb{R}^{n_\lambda}$ and $\boldsymbol{\mu} \in \mathbb{R}^{n_\mu}_{\geq 0}$ we have that*

$$V_N^0(\bar{x}) \geq D_N^\delta(\bar{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) - \delta \boldsymbol{\mu}^T \mathbf{d}.$$

Hence, a lower bound to V_N^0 is readily available from the optimization algorithm data in each iteration p. To compute an upper bound to the r.h.s. of (14) we define the following primal cost for initial condition $\bar{\mathbf{x}}$ using control trajectory $\mathbf{u}(0:N-1)$

$$P_N(\bar{\mathbf{x}}, \mathbf{u}(0:N-1)) = \begin{cases} \sum_{l=0}^{N-1} \ell(\boldsymbol{\xi}(l), \mathbf{u}(l)), & \text{if } \boldsymbol{\xi}(l+1) = \mathbf{A}\boldsymbol{\xi}(l) + \mathbf{B}\mathbf{u}(l), \\ & \quad \boldsymbol{\xi}(0) = \bar{\mathbf{x}}, \\ & \quad \boldsymbol{\xi}(l) \in \mathcal{X}, \mathbf{u}(l) \in \mathcal{U}, \\ & \quad l = 0, \dots, N-1 \\ \infty, & \text{else.} \end{cases}$$

The cost P_N is finite if the state and control trajectories are feasible and ∞ if they are not. Hence

$$P_N(\bar{\mathbf{x}}, \mathbf{u}(0:N-1)) \geq V_N(\bar{\mathbf{x}})$$

for any control trajectory $\mathbf{u}(0:N-1)$. The cost at iteration p in the algorithm is $P_N(\bar{\mathbf{x}}, \hat{\mathbf{u}}^p(0:N-1))$ where $\hat{\mathbf{u}}^p$ is extracted from $\mathbf{z}^p$ in Algorithm 1. By introducing the shifted control trajectory

$$\hat{\mathbf{u}}_s^p(0:N-1) = [\hat{\mathbf{u}}^p(1)^T, \dots, \hat{\mathbf{u}}^p(N-1)^T \; 0^T]^T$$

a cost for the next time step at iteration p is

$$P_N(\mathbf{A}\bar{\mathbf{x}} + \mathbf{B}\hat{\mathbf{u}}^p(0), \hat{\mathbf{u}}_s^p(0:N-1)).$$

We have that

$$P_N(\mathbf{A}\bar{\mathbf{x}}+\mathbf{B}\hat{\mathbf{u}}^p(0), \hat{\mathbf{u}}_s^p(0:N-1))+\ell(\bar{\mathbf{x}}, \hat{\mathbf{u}}^p(0)) \geq V_N(\mathbf{A}\bar{\mathbf{x}}+\mathbf{B}\hat{\mathbf{u}}^p(0))+\ell(\bar{\mathbf{x}}, \hat{\mathbf{u}}^p(0)) \tag{17}$$

which implies that an upper bound to the r.h.s. of (14) can readily be computed in any iteration p. The objective of the stopping condition is to adapt the constraint tightening δ and stop at iteration p such that

$$D_N^\delta(\bar{\mathbf{x}}, \boldsymbol{\lambda}^p, \boldsymbol{\mu}^p) - \delta(\boldsymbol{\mu}^p)^T \mathbf{d} \geq P_N(\mathbf{A}\bar{\mathbf{x}} + \mathbf{B}\hat{\mathbf{u}}^p(0), \hat{\mathbf{u}}_s^p(0:N-1)) + \alpha\ell(\bar{\mathbf{x}}, \hat{\mathbf{u}}^p(0)).$$

This implies feasibility of the next step due to the definition of P_N and that

$$V_N^0(\bar{\mathbf{x}}) \geq V_N^0(\mathbf{A}\bar{\mathbf{x}} + \mathbf{B}\hat{\mathbf{u}}^p(0)) + \alpha\ell(\bar{\mathbf{x}}, \hat{\mathbf{u}}^p(0))$$

holds due to Lemma 1 and (17). This, in turn, implies asymptotic stability and performance as specified by (15). In the stopping condition, there are three parameters that need to be set. These are the desired performance $\alpha \in (0,1)$, the initial constraint tightening $\delta_{\text{init}} \in (0,1)$, and an optimality tolerance $\epsilon \in [0.05, 0.001]$. The stopping condition is presented below.

Algorithm 2 ***Stopping condition***

Input*:* $\bar{\mathbf{x}}$
Set $p = 0$, $l = 0$, $\delta = \delta_{\text{init}}$
Run Δp *iterations of Algorithm 1 based on optimization problem* (16)
Do
  If $D_N^{\delta}(\bar{\mathbf{x}}, \boldsymbol{\lambda}^p, \boldsymbol{\mu}^p) \geq P_N(\bar{\mathbf{x}}, \hat{\mathbf{u}}^p(0:N-1)) - \frac{\epsilon}{l+1}\ell^*(\bar{\mathbf{x}})$
    or $\delta \mathbf{d}^T \boldsymbol{\mu}^p > \epsilon \ell^*(\bar{\mathbf{x}})$
      Set $\delta \leftarrow \delta/2$ *// reduce constraint tightening*
      Set $l \leftarrow l+1$ *// reduce optimality tolerance*
      Set $p = 0$ *// reset step size and iteration counter*
  End
  Run Δp *iterations of Algorithm 1 based on optimization problem* (16)
  Set $p \leftarrow p + \Delta p$
Until $D_N^{\delta}(\bar{\mathbf{x}}, \boldsymbol{\lambda}^p, \boldsymbol{\mu}^p) \geq P_N(\mathbf{A}\bar{\mathbf{x}} + \mathbf{B}\hat{\mathbf{u}}^p(0), \hat{\mathbf{u}}_s^p(0:N-1)) + \alpha\ell(\bar{\mathbf{x}}, \hat{\mathbf{u}}_0^p)$ **and**
  $\delta \mathbf{d}^T \boldsymbol{\mu}^p \leq \epsilon \ell^*(\bar{\mathbf{x}})$
Output*:* $\hat{\mathbf{u}}^p(0)$

To guarantee that the stopping condition will terminate with finite number of iterations and that the feasibility, stability, and performance results hold, there are restrictions on how the control horizon N must be chosen for a given $\alpha \in (0,1)$. The reader is referred to [8] for details of this choice.

3.5 Evaluation of Algorithm Efficiency

The efficiency of the algorithms presented in this chapter is evaluated by applying them to a randomly generated system. The cost function, constraints, and dynamics of the randomly generated system are specified in [4, Supplement A.1]. The system consists of three interconnected subsystems with five states and one input each, i.e., 15 states and 3 inputs in total. The magnitude of the largest eigenvalue is 1.1 and the upper and lower bounds on the states are chosen randomly from the intervals $[0.5, 1.5]$ and $[-0.15, 0.05]$ respectively. The upper and lower bounds on the inputs are chosen randomly from the interval $[0.5, 1.5]$ and $[-1.5, -0.5]$ respectively. The cost matrices $\mathbf{Q}$ and $\mathbf{R}$ are diagonal with diagonal elements chosen randomly from the interval $[1, 100]$. Using the method in [8] it is shown that using $\alpha = 0.01$, (14) is satisfied with control horizon $N = 6$.

In Table 1 the algorithm evaluation is presented. We compare the iteration complexity when using Algorithm 1 with a block-diagonal $\mathbf{L}$-matrix computed as in

(13). This distributed generalized accelerated gradient algorithm is referred to as Alg. 1, GAG. This is compared to when using Algorithm 1 with the optimal step size, i.e., with $\mathbf{L} = LI$ where L is the (smallest) Lipschitz constant to the dual function (8). This distributed accelerated gradient algorithm is referred to as Alg. 1, AG. We also compare these algorithms to when using a standard gradient method with the optimal step size, which is the traditional way of solving the dual problem in distributed fashion. This gradient method is obtained by setting $\bar{\mathbf{z}}_i^p = \mathbf{z}_i^p$, $\bar{\boldsymbol{\lambda}}_i^p = \boldsymbol{\lambda}_i^p$, and $\bar{\boldsymbol{\mu}}_i^p = \boldsymbol{\mu}_i^p$ in Algorithm 1. This distributed gradient method is referred to as Alg. 1, G. The methods are compared with and without the preconditioning presented in [7] and with and without the stopping condition in Algorithm 2. The comparison is made on 1000 randomly generated points that are chosen from a uniform distribution over the state constraint set. Also, an estimate of the region of attraction is presented and compared to the estimated region of attraction in centralized MPC where the terminal constraint set is chosen as the maximal output admissible set. The centralized MPC formulation is referred to as CMPC. All data in Table 1 that comes from Algorithm 1 are obtained by cold-starting the algorithm.

The first column in Table 1 specifies the algorithm used. The second and third columns specify the optimality tolerance ϵ and the initial constraint tightening δ_{init} in the stopping condition in Algorithm 2. If Algorithm 2 is not used, the parameters specify the following optimality condition

$$D_N^\delta(\bar{\mathbf{x}}, \boldsymbol{\lambda}^p, \boldsymbol{\mu}^p) \geq P_N(\bar{\mathbf{x}}, \hat{\mathbf{u}}^p(0:N-1)) - \epsilon \ell^*(\bar{\mathbf{x}}).$$

Columns four and five specify if preconditioning and the stopping condition in Algorithm 2 are used respectively. Columns six and seven specify the average and max number of iterations needed. The seventh column specifies the average final constraint tightening and the final columns specifies the estimated region of attraction, i.e., the percentage of initial conditions steered to the origin.

Table 1 reveals that by using accelerated gradient method instead of gradient methods, the algorithm is improved by a factor 20. Further, by using preconditioning an additional improvement by a factor 1-4 is achieved. The stopping condition, besides guaranteeing stability, improves the convergence by another factor 1-3. By allowing for $\mathbf{L}$-matrices instead of only scalar step sizes, we get another improvement by a factor around 10. Hence, using the presented methods, the number of iterations is, for this example, decreased by almost three orders of magnitude compared to the traditional and straightforward application of a gradient method, although the gradient method here is equipped with the optimal step size. Further, the comparison in estimated region of attraction for the presented method without a terminal constraint set and CMPC with a terminal constraint set reveals that the region of attraction can be increased significantly by not using a terminal constraint set.

Table 1 Algorithm comparison for Algorithm 1 based on the generalized accelerated gradient method (GAG), the accelerated gradient method (AG), and the gradient method (G). The algorithms are compared with and without preconditioning and the stopping condition in Algorithm 2. Average and max number of iterations as well as average constraint tightening is presented. Also, the region of attraction (R.o.A.) is estimated and compared to centralized MPC with a terminal constraint set.

Alg.	ϵ	δ_{init}	precond	stop cond	avg. # iters	max # iters	avg. δ	R.o.A.
Alg. 1, GAG	0.005	0.1	y	y	8.86	41	0.052	57.6 %
Alg. 1, GAG	0.005	0.01	y	n	19.36	47	0.01	57.6 %
Alg. 1, GAG	0.005	0.1	n	y	9.65	48	0.053	57.6 %
Alg. 1, GAG	0.005	0.01	n	n	25.00	53	0.01	57.6 %
Alg. 1, AG	0.005	0.1	y	y	64.92	122	0.054	57.6 %
Alg. 1, AG	0.005	0.01	y	n	146.68	400	0.01	57.6 %
Alg. 1, AG	0.005	0.1	n	y	126.68	455	0.053	57.6 %
Alg. 1, AG	0.005	0.01	n	n	515.30	1161	0.01	57.6 %
Alg. 1, G	0.005	0.1	y	y	1185.8	2906	0.055	57.6 %
Alg. 1, G	0.005	0.01	y	n	2458.9	10194	0.01	57.6 %
Alg. 1, G	0.005	0.1	n	y	3807.7	13991	0.089	57.6 %
Alg. 1, G	0.005	0.01	n	n	9954.9	21225	0.01	57.6 %
CMPC	-	-	-	-	-	-	-	0.7 %

4 Theoretical results

Application of accelerated gradient methods to DMPC based on dual decomposition and with an additional 1-norm term in the objective is presented in [6]. Iteration complexity bounds for the algorithm in [6] is presented in [7]. Based on the iteration complexity bounds, it is also shown how to precondition the optimization problem data optimally in [7]. The algorithm in [6] is in [3] extended to allow for step matrices instead of scalar step sizes. The algorithm in [3] is presented in Algorithm 1 in this chapter. Also, iteration complexity bounds to achieve a prespecified accuracy of the solution for Algorithm 1 are provided in [3]. Compared to existing methods that use dual decomposition to solve the optimization problem arising in distributed model predictive control, Algorithm 1 gives considerably lower iteration complexity.

In [8] it is shown for which initial conditions the stopping condition in Algorithm 2 is guaranteed to terminate. Also, a result guaranteeing feasibility, closed loop stability, and a prespecified performance when using the stopping condition in Algorithm 2 is presented. The region of attraction using the presented stopping condition compared to using a centralized MPC formulation with a terminal cost and a terminal constraint set, where the terminal constraint set is chosen as the maximal output admissible set, is evaluated in [8, 5]. For some systems, the region of attraction when using the presented stopping condition is significantly larger. Robustness to bounded errors when using the stopping condition in Algorithm 2 is shown in [5]. In [5] also an output feedback DMPC scheme based on the stopping condition in Algorithm 2 is presented.

5 Applications

The presented methods have been applied in a simulated hydro power valley benchmark problem [13]. The hydro power valley benchmark consists of six dams that are placed along a river and three lakes that are interconnected to the river. The dams are equipped with turbines to generate power and interconnections between the lakes and the river are equipped with pumps and turbines such that water can flow in any direction. The objective of the control problem is to follow a time-varying power reference with the power production, while keeping flows and water levels within allowed limits. The model contains nonlinearities and binary constraints that are addressed to enable for an efficient and well performing implementation of the methods presented in this chapter. The case study is presented in [2].

Another application where these methods have been used is for disturbance management in the process industry. Chemicals in the process industry are often manufactured in several production areas within a production site. The production areas are interconnected by the product flow and each interconnection usually has a buffer-tank. If problems occur in one of the production areas, the other production areas are affected. The objective of the control is to set the production rates for the individual production areas to maximize the total throughput while avoiding shut-down of any production area due to lack of product inflow, i.e., to avoid empty buffer-tanks. This case study is presented in [11].

Acknowledgements The authors were supported by the Swedish Research Council through the Linneaus center (LCCC).

References

1. Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sciences*, 2(1):183–202, October 2009.
2. M. D. Doan, P. Giselsson, T. Keviczky, B. De Schutter, and A. Rantzer. A distributed accelerated gradient algorithm for DMPC of a hydro power valley. Submitted to *Control Engineering Practice*. Available: `http://www.control.lth.se/Staff/PontusGiselsson.html`, 2012.
3. P. Giselsson. A generalized distributed accelerated gradient method for DMPC with iteration complexity bounds. Submitted to *2013 American Control Conference*. Available: `http://www.control.lth.se/Staff/PontusGiselsson.html`, 2012.
4. P. Giselsson. *Gradient-Based Distributed Model Predictive Control*. PhD thesis, Department of Automatic Control, Lund University, Sweden, November 2012.
5. P. Giselsson. Output feedback distributed model predictive control with inherent robustness properties. Submitted to *2013 American Control Conference*. Available: `http://www.control.lth.se/Staff/PontusGiselsson.html`, 2012.
6. P Giselsson, M. D. Doan, T. Keviczky, B. De Schutter, and A Rantzer. Accelerated gradient methods and dual decomposition in distributed model predictive control. *Automatica*, 2012. To appear. Available: `http://www.control.lth.se/Staff/PontusGiselsson.html`.

7. Pontus Giselsson. Optimal preconditioning and iteration complexity bounds for gradient-based optimization in model predictive control. Submitted to *2013 American Control Conference*. Available: `http://www.control.lth.se/Staff/PontusGiselsson.html`, 2012.
8. Pontus Giselsson and Anders Rantzer. On feasibility, stability and performance in distributed model predictive control. Submitted to *IEEE Transactions on Automatic Control*. Available: `http://www.control.lth.se/Staff/PontusGiselsson.html`, 2012.
9. Lars Grüne. Analysis and design of unconstrained nonlinear MPC schemes for finite and infinite dimensional systems. *SIAM Journal on Control and Optimization*, 48:1206–1228, 2009.
10. Lars Grüne and Anders Rantzer. On the infinite horizon performance of receding horizon controllers. *IEEE Transactions on Automatic Control*, 53:2100–2111, 2008.
11. A. Lindholm and P. Giselsson. Formulating an optimization problem for minimization of losses due to utilities. In *8th IFAC International Symposium on Advanced Control of Chemical Processes*, Singapore, 2012.
12. Yurii Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Netherlands, 1st edition, 2003.
13. C. Savorgnan and M. Diehl. Control benchmark of a hydro-power plant. Technical report, Hierarchical and Distributed Model Predictive Control for Large-Scale Systems (HD-MPC), 2011. Available at `http://www.ict-hd-mpc.eu/benchmarks/HD-MPC_hydropower_valley_public_benchmark.zip`.
14. Paul Tseng. On accelerated proximal gradient methods for convex-concave optimization. Submitted to SIAM J. Optim. Available: `http://www.math.washington.edu/~tseng/papers/apgm.pdf`, May 2008.
15. Wangmeng Zuo and Zhouchen Lin. A generalized accelerated proximal gradient approach for total-variation-based image restoration. *IEEE Transactions on Image Processing*, 20(10):2748–2759, October 2011.