Optimization and Engineering
https://doi.org/10.1007/s11081-020-09585-w

RESEARCH ARTICLE

Fast and stable nonconvex constrained distributed optimization: the ELLADA algorithm

Wentao Tang[1,2] · **Prodromos Daoutidis**[1]

Received: 4 May 2020 / Revised: 29 November 2020 / Accepted: 29 November 2020

Abstract
Distributed optimization using multiple computing agents in a localized and coordinated manner is a promising approach for solving large-scale optimization problems, e.g., those arising in model predictive control (MPC) of large-scale plants. However, a distributed optimization algorithm that is computationally efficient, globally convergent, amenable to nonconvex constraints remains an open problem. In this paper, we combine three important modifications to the classical alternating direction method of multipliers for distributed optimization. Specifically, (1) an extra-layer architecture is adopted to accommodate nonconvexity and handle inequality constraints, (2) equality-constrained nonlinear programming (NLP) problems are allowed to be solved approximately, and (3) a modified Anderson acceleration is employed for reducing the number of iterations. Theoretical convergence of the proposed algorithm, named ELLADA, is established and its numerical performance is demonstrated on a large-scale NLP benchmark problem. Its application to distributed nonlinear MPC is also described and illustrated through a benchmark process system.

Keywords Distributed optimization · Nonconvex optimization · Model predictive control · Acceleration

1 Introduction

Distributed optimization (Boyd et al. 2011) refers to methods of performing optimization using a distributed architecture—the monolithic problem is first decomposed into several subproblems, each handled by a corresponding solver (agent), and

✉ Prodromos Daoutidis
daou001@umn.edu

[1] Department of Chemical Engineering and Materials Science, University of Minnesota, Minneapolis, MN 55455, USA

[2] Present Address: Surface Operations, Projects and Technology, Shell Global Solutions (U.S.) Inc., Houston, TX 77082, USA

Published online: 03 January 2021

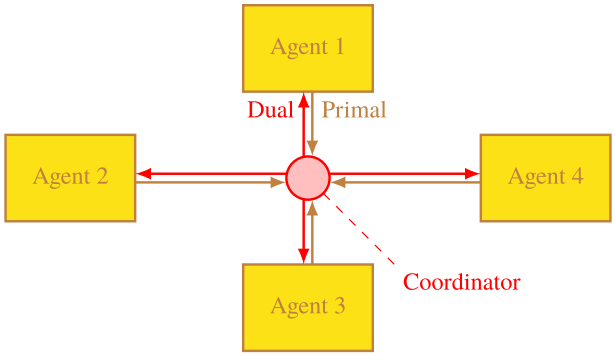

Fig. 1 Primal–dual distributed optimization

necessary information among the agents is communicated to coordinate the distributed computation. The alternating direction method of multipliers (ADMM), as the most classical and representative algorithm for distributed optimization, was proposed in the 1970s (Glowinski and Marroco 1975; Gabay and Mercier 1976) and has underdone significant development in the last decade. ADMM is an iterative primal–dual algorithm that is closely related to the method of multipliers (MM). In each iteration, the blocks of primal variables are alternately optimized, and the dual variables are updated according to the updated values of the primal variables. The underlying architecture of the algorithm is illustrated in Fig. 1, with two distinct groups of components—the distributed agents and the coordinator(s) collecting information about the primal and dual values from each other to solve their own subproblems.

Convergence is the most basic requirement of distributed optimization and also the central theme of a large amount of theoretical research on ADMM. For convex problems and nonconvex problems whose nonconvexity resides in the objective function, by connecting the ADMM algorithm with the monotone operator theory and Douglas–Rachford splitting, the Fejér monotonicity and monotonicity of the augmented Lagrangian become standard arguments of proving theoretical convergence guarantees (He and Yuan 2012; Nishihara et al. 2015; Hong et al. 2016; Hong and Luo 2017; Wang et al. 2019; Themelis and Patrinos 2020). However, nonconvex constraints appear much more difficult to handle. To guarantee convergence, Hours and Jones (2015) suggested dualizing and penalizing all nonconvex constraints, making them undifferentiated and tractable by ADMM; however, this alteration of the problem structure eliminates the option for distributed agents to use any subroutine other than MM. Houska et al. (2016) used a quadratic programming problem to decide the dual variables in the augmented Lagrangian as well as an extrapolation of primal updates; this algorithm, however, involves a central agent that extracts Hessian and gradient information of the subsystem models from the distributed agents in every iteration, and is thus essentially semi-centralized. Scutari et al. (2016) adopted feasibility-preserving convex approximations to approach the solution, which is applicable to problems without nonconvex equality constraints. We note that several

recent papers (Sun and Sun 2019; Jiang et al. 2019; Yang et al. 2020) proposed the idea of placing slack variables corresponding to the inter-subsystem constraints and forcing the decay to zero by tightening the penalty parameters of slack variables. This modification to the ADMM with slack variables and their penalties leads to a globally convergent *extra-layer* augmented Lagrangian-based algorithm with preserved agent-coordinator problem architecture.

Computational efficiency is also of critical importance for distributed optimization. The slothfulness of primal–dual algorithms typically arises from two issues. First, the subgradient (first-order) update of dual variables restricts the number of iterations to be of linear complexity (Hong and Luo 2017). For convex problems, momentum methods (Goldstein et al. 2014; Ouyang et al. 2015) or Krylov subspace methods (Zhang and White 2018) can be adopted for acceleration, which, however, do not directly extend to nonconvex problems. Under nonconvexity, it was only very recently realized that *Anderson acceleration*, a multi-secant technique for fixed-point problems, can be generally used to accelerate the dual variables (Zhang et al. 2018, 2019; Fu et al. 2019). The second cause for the high computational cost of distributed optimization is the instruction on the distributed agents to fully solve their subproblems to high precision in each iteration. Such exhaustive efforts may be unnecessary since the dual information to be received from the coordinator will keep changing. For convex problems, it is possible to linearize the augmented Lagrangian and replace the distributed subproblems with Arrow–Hurwicz–Uzawa gradient flows (Dhingra et al. 2019). In the presence of nonconvexity of the objective functions, a dual perturbation technique to restore the convergence of the augmented Lagrangian was proposed (Hajinezhad and Hong 2019). It is yet unknown how to accommodate such gradient flows to nonconvex constraints. A different approach is to allow inexact solution of the subproblems with adaptively tightening tolerances (Eckstein and Yao 2017, 2018). Such an *approximate ADMM algorithm* allows a better balance between the primal and dual updates, and avoids wasteful computational steps inside the subroutines.

An important engineering application that demands both a convergent and efficient distributed optimization algorithm is model predictive control (MPC), which is an advanced control strategy widely adopted in process industries. In MPC, control decisions are made through solving an optimal control problem minimizing the cost associated with the predicted trajectory in a future horizon subject to the system dynamics and operational constraints (Rawlings et al. 2017). For large-scale systems, it is desirable to seek a decomposition, e.g., using community detection or network block structures (Daoutidis et al. 2018, 2019; Tang et al. 2018), and deploy distributed MPC strategies (Scattolini 2009; Christofides et al. 2013), which promises better performance than fully decentralized MPC by enabling coordination, while avoiding assembling and computing on a monolithic model as in centralized MPC. In general, distributed nonlinear MPC with subsystem interactions should be considered as a *distributed optimization problem under nonconvex constraints*. Whether the distributed optimization algorithm converges affects the quality of control decisions and hence the closed-loop control performance, whereas computational efficiency determines whether a distributed nonlinear MPC strategy can be practically implemented online.

Although ADMM for distributed MPC has been discussed (Farokhi et al. 2014; Mota et al. 2014), due to the lack of convergent and efficient nonconvex constrained distributed optimization algorithms, its application is so far limited mainly to linear systems (Giselsson et al. 2013; Wang and Ong 2017). The most common approach of distributed nonlinear MPC is to iterate the control inputs among the subsystems (in sequence or in parallel) (Stewart et al. 2010; Liu et al. 2010; Chen et al. 2012). The input iteration routine is typically either semi-decentralized by implicitly assuming that the subsystems interact only through inputs and considering state coupling as disturbances, or semi-centralized by using moving-horizon predictions based on the entire system, which, however, contradicts the fact that the subsystem models should be usually packaged inside the local agents rather than shared over the entire system. In a different vein, efficient centralized MPC algorithms have been proposed which exploit the underlying sparse patterns in the variable-constraint structure of the optimal control problems and decompose the linear algebraic operations involved in the optimization solver (Wang and Boyd 2009; Patterson and Rao 2014; Biegler and Thierry 2018). While these efficient centralized MPC algorithms improve computational efficiency without compromising the solution quality, distributed MPC is still advantageous in its capability of solving very large MPC problems on a localized subsystem basis without the need of engineering the solver details. Moreover, such structure exploitation can also be utilized by the solvers employed in distributed MPC. The crucial computational issue faced by distributed MPC is its iterative complexity and this will be addressed in this paper.

The purpose of this work is to develop a convergent and computationally efficient algorithm for distributed optimization under nonconvex constraints. Although the algorithm is in principle not restricted to any specific problem, we consider distributed nonlinear MPC as an application of special interest. Based on the above discussion, we identify the following modifications to the classical ADMM algorithm as the key to mitigating the challenges in convergence and computational complexity: (1) additional slack variables are placed on the constraints relating the distributed agents and the coordinator, (2) approximate optimization is performed in the distributed agents, and (3) the Anderson acceleration technique is adopted by the coordinator. We therefore combine and extend as appropriate these techniques into a new algorithm with a two-layer augmented Lagrangian-based architecture, in which the outer layer handles the slack variables as well as inequality constraints by using a barrier technique, and the inner layer performs approximate ADMM under an acceleration scheme. With guaranteed stability and elevated speed, *to the best knowledge of the authors, the proposed algorithm is the first practical and generic algorithm of its kind for distributed nonlinear MPC with truly localized model information.* We name this algorithm as **ELLADA** (standing for **e**xtra-**l**ayer augmented **L**agrangian-based **a**ccelerated **d**istributed **a**pproximate optimization).

The paper discusses the motivation, develops the ELLADA algorithm and establishes its theoretical properties, and illustrates its application through case studies. The remainder of this paper is organized as follows. In Sect. 2, we first review the classical ADMM and its modified versions. Then we derive our ELLADA algorithm in Sect. 3 with a trilogy pattern. First, a basic two-layer augmented Lagrangian-based algorithm (ELL) is introduced and its convergence is discussed. Then the approximate solution of

equality-constrained NLP problems and the Anderson acceleration scheme are incorporated to form the ELLA and ELLADA algorithms. In Sect. 4, a large-scale nonlinear optimization benchmark problem is used to demonstrate the numerical performance of the proposed ELLADA algorithm. The implementation of ELLADA on the distributed optimization problem involved in distributed nonlinear MPC is shown in Sect. 5, and a case study on a benchmark process is examined in Sect. 6. Conclusions and discussions are given in Sect. 7.

2 ADMM and its modifications

2.1 ADMM

The alterating direction method of multipliers is the most commonly used algorithm for distributed optimization under linear equality constraints (Boyd et al. 2011). Specifically, consider the following problem

$$\min_{x,\bar{x}} \ f(x) + g(\bar{x}) \quad \text{s.t.} \quad Ax + B\bar{x} = 0 \tag{1}$$

with two blocks of variables x and $\bar{x}$, where f and g are usually assumed to be convex. [The symbols in (1) are not related to the ones in Sect. 5.] The augmented Lagrangian for such a constrained optimization problem is

$$L(x, \bar{x}; y) = f(x) + g(\bar{x}) + y^{\top}(Ax + B\bar{x}) + \frac{\rho}{2}\|Ax + B\bar{x}\|^2, \tag{2}$$

in which y stands for the vector of dual variables (Lagrangian multipliers) and $\rho > 0$ is called the penalty parameter. According to the duality theory, the optimal solution should be determined by a saddle point of the augmented Lagrangian:

$$\sup_{y} \min_{x,\bar{x}} \ L(x, \bar{x}; y). \tag{3}$$

The classical method of multipliers (MM) deals with this saddle point problem with an iterative procedure, where the primal variables are optimized first and then the dual variables are updated with a subgradient ascent (Bertsekas 2016, Chapter 6):

$$\begin{aligned} (x^{k+1}, \bar{x}^{k+1}) &= \arg\min_{x,\bar{x}} L\left(x, \bar{x}; y^k\right), \\ y^{k+1} &= y^k + \rho\left(Ax^{k+1} + B\bar{x}^{k+1}\right), \end{aligned} \tag{4}$$

in which the superscript stands for the count of iterations. In a distributed context, x and $\bar{x}$ usually can not be optimized simultaneously. ADMM is thus an approximation of MM that allows the optimization of x and $\bar{x}$ to be performed separately, i.e.,

$$\begin{aligned} x^{k+1} &= \arg\min_{x} L\left(x, \bar{x}^k; y^k\right), \\ \bar{x}^{k+1} &= \arg\min_{\bar{x}} L\left(x^k, \bar{x}; y^k\right), \\ y^{k+1} &= y^k + \rho\left(Ax^{k+1} + B\bar{x}^{k+1}\right). \end{aligned} \tag{5}$$

Since the appearance of ADMM in 1970s (Glowinski and Marroco 1975; Gabay and Mercier 1976), there have been many works regarding its theoretical properties, extensions and applications. As we have mentioned in the Introduction, ADMM is known to have a linear convergence rate for convex problems. This does not change when the variables are constrained in convex sets. For example, if $x \in \mathcal{X}$, it suffices to modify the corresponding term $f(x)$ in the objective function by adding an indicator function $\mathbb{I}_{\mathcal{X}}(x)$ (equal to 0 if $x \in \mathcal{X}$ and $+\infty$ otherwise), which is still a convex function.

2.2 ADMM with approximate updates

Unless the objective terms $f(x)$ and $g(\bar{x})$ are of simple forms such as quadratic functions, the optimization of x and $\bar{x}$ in (5) does not have an exact solution. Usually, iterative algorithms for nonlinear programming need to be called for the first two lines of (5), and always searching for a highly accurate solution in each ADMM iteration will result in an excessive computational cost. It is thus desirable to solve the optimization subproblems in ADMM inexactly when the dual variables are yet far from the optimum, i.e., to allow x^{k+1} and $\bar{x}^{k+1}$ to be chosen such that

$$d_x^{k+1} \in \partial_x L\left(x^{k+1}, \bar{x}^k; y^k\right), \quad d_{\bar{x}}^{k+1} \in \partial_{\bar{x}} L\left(x^{k+1}, \bar{x}^{k+1}; y^k\right), \tag{6}$$

where ∂_x and $\partial_{\bar{x}}$ represent the subgradients with respect to x and $\bar{x}$, respectively, and d_x and $d_{\bar{x}}$ are not exactly 0 but only converging to 0 asymptotically. For example, one can assign externally a shrinking and summable sequence of absolute errors (Eckstein and Bertsekas 1992):

$$\|d_x^k\| \le \epsilon_x^k, \|d_{\bar{x}}^k\| \le \epsilon_{\bar{x}}^k, \sum_{k=1}^{\infty} \epsilon_x^k < \infty, \sum_{k=1}^{\infty} \epsilon_{\bar{x}}^k < \infty, \tag{7}$$

or a sequence of relative errors to the errors proportional to other variations in the algorithm (Eckstein and Yao 2017; Xie et al. 2017).

It was shown in Eckstein and Yao (2017) that a relative error criterion for terminating the iterations in subproblems, compared to other approximation criteria such as a summable absolute error sequence, better reduces the total number of subroutine iterations throughout the ADMM algorithm. Such a relative error criterion is a *constructive* one, rendered to guarantee the decrease of a quadratic distance between the intermediate solutions $(x^k, \bar{x}^k, y^k)$ and the optimum $(x^*, \bar{x}^*, y^*)$. In the context of distributed optimization problems under nonconvex constraints, since the convergence proof is established on a different basis from the quadratic distance, the construction of such a criterion must be reconsidered. We will address this issue in Sect. 3.2.

2.3 Anderson acceleration

Linear convergence of the classical ADMM is essentially the result of subgradient dual update, which uses the information of only the first-order derivatives with respect to the dual variables: $\partial_y L = Ax + B\bar{x}$. The idea of creating a *quadratically convergent* algorithm using only first-order derivatives originates back from Nesterov's approach of solving convex optimization problems, which performs iterations based on a linear extrapolation of the previous two iterations instead of the current solution alone (Nesterov 1983). Such a *momentum* method can be used to accelerate the ADMM algorithm, which can be seen as iterations over the second block of primal variables $\bar{x}$ and the dual variables y (Goldstein et al. 2014). However, such a momentum is inappropriate for nonconvex problems, since the behavior of the extrapolated point can not be well controlled by a bound on the curvature of the objective function.

Therefore, we resort to a different type of technique—Anderson acceleration, which was proposed in Anderson (1965) first and later "rediscovered" in the field of chemical physics (Pulay 1980). Generally speaking, Anderson acceleration is used to solve the fixed-point iteration problem

$$w = h_0(w) \tag{8}$$

for some vector w and non-expansive mapping h_0 (satisfying $\|h_0(w) - h_0(w')\| \leq \|w - w'\|$ for any w and w'). Different from the simple Krasnoselskii–Mann iteration $w^{k+1} = \kappa w^k + (1-\kappa)h_0(w^k)$ ($\kappa \in (0,1)$), Anderson acceleration takes a quasi-Newton approach, which aims at a nearly quadratic convergence rate (Fang and Saad 2009). Specifically,[1] in each iteration k, the results from the previous m iterations are recalled from memory to form the matrix of secants in w and $h(w) = w - h_0(w)$:

$$\begin{aligned} \Delta_w^k &= \left[\delta_w^{k-m} \ \ldots \ \delta_w^{k-1}\right], \delta_w^{k'} = w^{k'+1} - w^{k'}, k' = k-m, \ldots, k-1; \\ \Delta_h^k &= \left[\delta_h^{k-m} \ \ldots \ \delta_h^{k-1}\right], \delta_h^{k'} = h\left(w^{k'+1}\right) - h\left(w^{k'}\right), k' = k-m, \ldots, k-1. \end{aligned} \tag{9}$$

An estimated Jacobian is given by

$$H_k = I + \left(\Delta_h^k - \Delta_w^k\right)\left(\Delta_w^{k\top}\Delta_w^k\right)^{-1}\Delta_w^{k\top}, \tag{10}$$

or

$$H_k^{-1} = I + \left(\Delta_w^k - \Delta_h^k\right)\left(\Delta_w^{k\top}\Delta_h^k\right)^{-1}\Delta_w^{k\top}, \tag{11}$$

which minimizes the Frobenius norm of $B_k - I$ subject to $B_k\Delta_w^k = \Delta_h^k$. Then the quasi-Newton iteration $w^{k+1} = w^k - H_k^{-1}h^k$ leads to a weighted sum of the previous m function values:

[1] There are two different types of Anderson acceleration. Here we focus on Type I, which was found to have better performance (Fang and Saad 2009) and was improved in Zhang et al. (2018).

$$w^{k+1} = \sum_{m'=0}^{m} \alpha_{m'}^{k} h_0\left(x^{k-m+m'}\right) \tag{12}$$

where the weights $\left\{\alpha_{m'}^{k}\right\}_{m'=0}^{m}$ are specified by

$$\alpha_{m'}^{k} = \begin{cases} s_0^k, & m' = 0 \\ s_{m'}^k - s_{m'-1}^k, & m' = 1, \ldots, m-1 \\ 1 - s_{m-1}^k, & m' = m \end{cases}, \tag{13}$$

with $s_{m'}^k$ being the m'th component s^k:

$$s^k = (\Delta_w^{k\top} \Delta_h^k)^{-1} \Delta_w^{k\top} h^k. \tag{14}$$

Anderson acceleration (12) may not always be convergent, although local convergence was studied in some special cases (Toth and Kelley 2015). Recently, a globally convergent modification of Anderson acceleration was proposed in Zhang et al. (2018), where regularization, restarting, and safeguarding measures are taken to ensure the well-conditioning of the Δ_w^k matrix, boundedness of the inverse Jacobian estimate (11), and acceleration only in a safety region, respectively.

The relevance of Anderson acceleration to ADMM lies in that the ADMM algorithm (5) can be seen as fixed-point iterations $(\bar{x}^k, y^k) \to (\bar{x}^{k+1}, y^{k+1})$, $k = 0, 1, 2, \ldots$ (Zhang et al. 2019), which is the same idea underlying the ADMM with Nesterov acceleration. For problems with nonconvex constraints, the iteration mapping h is not necessarily non-expansive, and hence one can not directly establish the convergence of Anderson acceleration with the original techniques used in Zhang et al. (2018). We will address this issue in Sect. 3.3.

2.4 ADMM under nonconvex constraints

The presence of nonconvexity largely increases the difficulty of distributed optimization. Most of the work in nonconvex ADMM considers problems with nonconvex objective function with bounded Hessian eigenvalues or the Kurdyka–Łojasiewicz property assumptions, under which a convergence rate of $\mathcal{O}\left(1/\sqrt{k}\right)$ (slower than that of convex ADMM, $\mathcal{O}(1/k)$) was established (Li and Pong 2015; Hong et al. 2016; Wang et al. 2019). However, for many distributed optimization problems, e.g., the distributed MPC of nonlinear processes, there exist nonconvex constraints on the variables, which is intrinsically non-equivalent to the problems with nonconvex objective functions. For our problem of interest, the relevant works are scarce.

Here we introduce the algorithm of Sun and Sun (2019) for (1) under nonconvex constraints $x \in \mathcal{X}$ and $\bar{x} \in \bar{\mathcal{X}}$, reformulated with slack variables z:

$$\min_{x,\bar{x},z} \; f(x) + g(\bar{x}) \quad \text{s.t. } Ax + B\bar{x} + z = 0, z = 0, \; x \in \mathcal{X}, \; \bar{x} \in \bar{\mathcal{X}}. \tag{15}$$

The augmented Lagrangian is now written as

$$L(x,\bar{x},z;y,\lambda,\rho,\beta)=f(x)+g(\bar{x})+\mathbb{I}_{\mathcal{X}}(x)+\mathbb{I}_{\bar{\mathcal{X}}}(\bar{x})+y^{\top}(Ax+B\bar{x}+z)+\frac{\rho}{2}\|Ax+B\bar{x}+z\|^2+\lambda^{\top}z+\frac{\beta}{2}\|z\|^2. \tag{16}$$

The algorithm is a two-layer one, where each outer iteration (indexed by k) contains a series of inner iterations (indexed by r). In the inner iterations, the classical ADMM algorithm is used to update x, $\bar{x}$, z and y in sequence, while keeping λ and β unchanged:

$$\begin{aligned}
x^{k,r+1}&=\arg\min_{x} L\left(x,\bar{x}^{k,r},z^{k,r};y^{k,r},\lambda^k,\rho^k,\beta^k\right)\\
&=\arg\min_{x\in\mathcal{X}} f(x)+\frac{\rho^k}{2}\left\|Ax+B\bar{x}^{k,r}+z^{k,r}+\frac{y^{k,r}}{\rho^k}\right\|^2\\
\bar{x}^{k,r+1}&=\arg\min_{\bar{x}} L\left(x^{k,r+1},\bar{x},z^{k,r};y^{k,r},\lambda^k,\rho^k,\beta^k\right)\\
&=\arg\min_{\bar{x}\in\bar{\mathcal{X}}} g(\bar{x})+\frac{\rho^k}{2}\left\|Ax^{k,r+1}+B\bar{x}+z^{k,r}+\frac{y^{k,r}}{\rho^k}\right\|^2\\
z^{k,r+1}&=\arg\min_{z} L\left(x^{k,r+1},\bar{x}^{k,r+1},z;y^{k,r},\lambda^k,\rho^k,\beta^k\right)\\
&=-\frac{\rho^k}{\rho^k+\beta^k}\left(Ax^{k,r+1}+B\bar{x}^{k,r+1}+\frac{y^{k,r}}{\rho^k}\right)-\frac{1}{\rho^k+\beta^k}\lambda^k\\
y^{k,r+1}&=y^{k,r}+\rho^k\left(Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r+1}\right)
\end{aligned} \tag{17}$$

Under mild assumptions, in the presence of slack variables z, it was proved (Sun and Sun 2019) that if one chooses $\rho^k=2\beta^k$, then the inner iterations converge to the set of stationary points $\left(x^k,\bar{x}^k,z^k,y^k\right)$ of the relaxed problem

$$\begin{aligned}
\min_{x,\bar{x},z}\quad & f(x)+g(\bar{x})+\lambda^{k\top}z+\frac{\beta^k}{2}\|z\|^2\\
\text{s.t.}\quad & Ax+B\bar{x}+z=0, x\in\mathcal{X},\bar{x}\in\bar{\mathcal{X}}.
\end{aligned} \tag{18}$$

Then in the outer iterations, the dual variables λ^k are updated. To enforce the convergence of the slack variables to zero, the corresponding penalty β^k is amplified by a ratio $\gamma>1$ if the returned z^k from the inner iterations does not decay enough from the previous outer iteration z^{k-1} ($\|z^k\|>\omega\|z^{k-1}\|, \omega\in(0,1)$). The outer iteration is written as

$$\lambda^{k+1}=\prod_{\left[\underline{\lambda},\overline{\lambda}\right]}\left(\lambda^k+\beta^k z^k\right),\quad \beta^{k+1}=\begin{cases}\gamma\beta^k, & \|z^k\|>\omega\|z^{k-1}\|\\ \beta^k, & \|z^k\|\le\omega\|z^{k-1}\|\end{cases} \tag{19}$$

in which the projection Π onto a predefined compact hypercube $\left[\underline{\lambda},\overline{\lambda}\right]$ is used to guarantee the boundedness of the dual variables and hence the augmented Lagrangian L. If the augmented Lagrangian L remains bounded despite the increase of the penalty parameters ρ^k and β^k, the algorithm converges to a stationary point of the original problem (1). The iterative complexity of such an algorithm to reach an ϵ -approximate stationary point is $\mathcal{O}\left(\epsilon^{-4}\ln\left(\epsilon^{-1}\right)\right)$.

In the next section, building on the algorithm of Sun and Sun (2019) that guarantees the convergence of distributed optimization under nonconvex constraints, we propose a new algorithm that integrates into it the ideas of approximate ADMM and Anderson acceleration, aiming at improving the computational efficiency.

3 Proposed algorithm

3.1 Basic algorithm and its convergence

Consider an optimization problem in the following form:

$$\begin{aligned} \min_{x,\bar{x}} \quad & f(x) + g(\bar{x}) \\ \text{s.t.} \quad & Ax + B\bar{x} = 0, \\ & x \in \mathcal{X} = \{x | \phi(x) \leq 0, \psi(x) = 0\}, \bar{x} \in \bar{\mathcal{X}} \end{aligned} \tag{20}$$

or equivalently with slack variables

$$\begin{aligned} \min_{x,\bar{x},z} \quad & f(x) + g(\bar{x}) \\ \text{s.t.} \quad & Ax + B\bar{x} + z = 0, z = 0, \\ & x \in \mathcal{X} = \{x | \phi(x) \leq 0, \psi(x) = 0\}, \bar{x} \in \bar{\mathcal{X}}. \end{aligned} \tag{21}$$

We make the following assumptions.

Assumption 1 Assume that f is lower bounded, i.e., $\exists \underline{f}$ such that $\forall x \in \mathcal{X}$, $f(x) \geq \underline{f}$.

Assumption 2 Function g is lower bounded.

Our basic algorithm (Algorithm 1) for (21) is slightly modified from the procedure of Sun and Sun (2019), which considered the case where $g(x) = 0$ and $\bar{\mathcal{X}}$ is a hypercube. The algorithm uses an inner loop of ADMM iterations and an outer loop of MM with possibly amplifying penalty parameters. The inner iterations are terminated when the following criterion is met

$$\begin{aligned} \epsilon_1^k \geq \epsilon_1^{k,r} &:= \left\| \rho^k A^\top (B\bar{x}^{k,r+1} + z^{k,r+1} - B\bar{x}^{k,r} - z^{k,r}) \right\|, \\ \epsilon_2^k \geq \epsilon_2^{k,r} &:= \left\| \rho^k B^\top (z^{k,r+1} - z^{k,r}) \right\|, \\ \epsilon_3^k \geq \epsilon_3^{k,r} &:= \left\| Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1} \right\|. \end{aligned} \tag{22}$$

Set: *Bound of dual variables* $\left[\underline{\lambda}, \overline{\lambda}\right]$, *shrinking ratio of slack variables* $\omega \in [0, 1)$, *amplifying ratio of penalty parameter* $\gamma > 1$, *diminishing outer iteration tolerances* $\left\{\epsilon_1^k, \epsilon_2^k, \epsilon_3^k\right\}_{k=1}^{\infty} \downarrow 0$, *terminating tolerances* $\epsilon_1, \epsilon_2, \epsilon_3 > 0$;

Initialization: *Starting points* x^0, $\bar{x}^0$, z^0, *duals and bounds* $\lambda^1 \in \left[\underline{\lambda}, \overline{\lambda}\right]$, *penalty* $\beta^1 > 0$;

Outer iteration count $k \leftarrow 0$;

while *stationarity criterion* (25) *is not met* **do**

  $\rho^k = 2\beta^k$;

  Inner iteration count $r \leftarrow 0$;

  Initialization: $x^{k,0}$, $\bar{x}^{k,0}$, $z^{k,0}$, $y^{k,0}$ *satisfying* $\lambda^k + \beta^k z^{k,0} + y^{k,0} = 0$;

  while *stopping criterion* (22) *is not met* **do**

    $x^{k,r+1} = \arg\min_{x \in \mathcal{X}} f(x) + \frac{\rho^k}{2} \left\| Ax + B\bar{x}^{k,r} + z^{k,r} + \frac{y^{k,r}}{\rho^k} \right\|^2$;

    $\bar{x}^{k,r+1} = \arg\min_{\bar{x} \in \bar{\mathcal{X}}} g(\bar{x}) + \frac{\rho^k}{2} \left\| Ax^{k,r+1} + B\bar{x} + z^{k,r} + \frac{y^{k,r}}{\rho^k} \right\|^2$;

    $z^{k,r+1} = -\frac{\rho^k}{\rho^k + \beta^k} \left(Ax^{k,r+1} + B\bar{x}^{k,r+1} + \frac{y^{k,r}}{\rho^k} \right) - \frac{1}{\rho^k + \beta^k} \lambda^k$;

    $y^{k,r+1} = y^{k,r} + \rho^k \left(Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1} \right)$;

    $r \leftarrow r + 1$;

  end

  $\left(x^{k+1}, \bar{x}^{k+1}, z^{k+1}, y^{k+1}\right) \leftarrow \left(x^{k,r}, \bar{x}^{k,r}, z^{k,r}, y^{k,r}\right)$;

  Update λ^{k+1} and β^{k+1} by (19);

end

Lemma 1 (Convergence of inner iterations) *Suppose that Assumptions 1 and 2 hold. When* $\rho^k = 2\beta^k$, *the inner iterations are terminated at a finite r when* (22) *is met and the point* $\left(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r+1}\right)$ *satisfies the following conditions*

$$\begin{aligned} d_1^k &\in \partial f\left(x^{k,r+1}\right) + \mathcal{N}_{\mathcal{X}}\left(x^{k,r+1}\right) + A^\top y^{k,r+1} \\ d_2^k &\in \partial g\left(\bar{x}^{k,r+1}\right) + \mathcal{N}_{\bar{\mathcal{X}}}\left(\bar{x}^{k,r+1}\right) + B^\top y^{k,r+1} \\ 0 &= \lambda^k + \beta^k z^{k,r+1} + y^{k,r+1}, \quad d_3^k = Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1} \end{aligned} \tag{23}$$

for some d_1^k, d_2^k *and* d_3^k *satisfying* $\left\|d_1^k\right\| \le \epsilon_1^k$, $\left\|d_2^k\right\| \le \epsilon_2^k$ *and* $\left\|d_3^k\right\| \le \epsilon_3^k$, *respectively.* $\mathcal{N}_{\mathcal{X}}(x)$ $(\mathcal{N}_{\bar{\mathcal{X}}}(\bar{x}))$ *refers to the normal cone to the set* $\mathcal{X}$ $(\bar{\mathcal{X}})$ *at point* x $(\bar{x})$:

$$\mathcal{N}_{\mathcal{X}}(x) = \left\{v | v^\top \left(x' - x\right) \le 0, \forall x' \in \mathcal{X}\right\}. \tag{24}$$

The proof of the above lemma is given in "Appendix 1" using the augmented Lagrangian (16) as a decreasing Lyapunov function throughout the inner iterations (Hong et al. 2016; Hong and Luo 2017), which gives the convergence of the inner iterations. It is apparent that if $\epsilon_1^k, \epsilon_2^k, \epsilon_3^k$ are all equal to 0, (23) is the Karush–Kuhn–Tucker optimality condition of the relaxed problem (18) (Rockafellar and Wets 1998). To establish the convergence of outer iterations, we need to make the following assumption to restrict the upper level of the augmented Lagrangian.

Assumption 3 The augmented Lagrangians are uniformly upper bounded at initialization of all inner iterations, i.e., there exists $\overline{L} \geq L\left(x^{k,0}, \bar{x}^{k,0}, z^{k,0}, y^{k,0}, \lambda^k, \rho^k, \beta^k\right)$ for all k.

The above assumption is actually a "warm start" requirement. Suppose that we have a feasible solution $\left(x^0, \bar{x}^0\right)$ to the original problem (20), then we can always choose $x^{k,0} = x^0$, $\bar{x}^{k,0} = \bar{x}^0$, $z^{k,0} = 0$, $y^{k,0} = -\lambda^k$ to guarantee an $\overline{L} = f\left(x^0\right) + g\left(\bar{x}^0\right)$.

Lemma 2 (Convergence of outer iterations) *Suppose that Assumptions 1, 2 and3 hold. Then for any ϵ_1, ϵ_2, and $\epsilon_3 > 0$, within a finite number of outer iterations k, Algorithm 1 finds an approximate stationary point $\left(x^{k+1}, \bar{x}^{k+1}, z^{k+1}, y^{k+1}\right)$ of (20), satisfying*

$$\begin{aligned} d_1 &\in \partial f\left(x^{k+1}\right) + \mathcal{N}_{\mathcal{X}}\left(x^{k+1}\right) + A^\top y^{k+1} \\ d_2 &\in \partial g\left(\bar{x}^{k+1}\right) + \mathcal{N}_{\bar{\mathcal{X}}}\left(\bar{x}^{k+1}\right) + B^\top y^{k+1} \\ d_3 &= Ax^{k+1} + B\bar{x}^{k+1} \end{aligned} \tag{25}$$

for some d_1, d_2, d_3 satisfying $\left\|d_j\right\| \leq \epsilon_j$, $j = 1, 2, 3$.

See "Appendix 2" for a proof. In addition to the convergence, we can also establish a theoretical complexity. Previously in Sun and Sun (2019), it was shown that to reach an ϵ-approximate stationary point satisfying (25) with $\epsilon_1, \epsilon_2, \epsilon_3 = \epsilon > 0$, the total number of inner iterations needed is of the order $\mathcal{O}\left(\epsilon^{-4} \ln(1/\epsilon)\right)$. Here, we show that by appropriately choosing the way that the tolerances $\left(\epsilon_1^k, \epsilon_2^k, \epsilon_3^k\right)$ shrink, the iteration complexity can be provably reduced anywhere in $\left[\mathcal{O}\left(\epsilon^{-2}\right), \mathcal{O}\left(\epsilon^{-4}\right)\right]$, for which a proof is given in "Appendix 3".

Lemma 3 (Complexity of the basic algorithm) *Suppose that Assumptions 1, 2 and3 hold. For some constant $\vartheta \in (0, \omega]$, choose $\epsilon_1^k \sim \mathcal{O}\left(\vartheta^k\right)$, $\epsilon_2^k \sim \mathcal{O}\left(\vartheta^k\right)$, and $\epsilon_3^k \sim \mathcal{O}\left((\vartheta/\beta)^k\right)$. Then each outer iteration k requires $R^k \sim \mathcal{O}\left((\vartheta\omega)^{-2k}\right)$ inner iterations. Hence, for the Algorithm 1 to reach an ϵ-approximate stationary point, the total iterations needed* is $R \sim \mathcal{O}\left(\epsilon^{-2(1+\varsigma)}\right)$, *where $\varsigma = \log_\vartheta \omega \in (0, 1]$.*

3.2 Approximate algorithm

We note that the basic algorithm requires undesirable complete minimization of x and $\bar{x}$ in each inner iteration (Lines 9–10, Algorithm 1). For simplicity, we assume that such a minimization oracle[2] exists for $\bar{x}$.

[2] We use the word "oracle" with its typical meaning in mathematics and computer science. An oracle refers to an ad hoc numerical or computational procedure, regarded as a black box mechanism, to generate the needed results as its outputs based on some input information.

Assumption 4 The minimization of the augmented Lagrangian with respect to $\bar{x}$ (Line 10, Algorithm 1) admits a unique explicit solution

$$\bar{x}^{k,r+1} = G\left(B, Ax^{k,r+1} + z^{k,r} + y^{k,r}/\rho^k, \rho^k\right). \tag{26}$$

Let us also assume that the problem has a smoothness property as follows.

Assumption 5 Functions f, ϕ and ψ are continuously differentiable, and $\mathcal{X}$ has a nonempty interior.

Under this smoothness assumption, the KKT condition for x-minimization is written as the following equalities with $\mu \geq 0$ and ν representing the Lagrangian dual variables corresponding to the inequalities $\phi(x) \leq 0$ and $\psi(x) = 0$, respectively

$$\begin{aligned} 0 &= \nabla f\left(x^{k,r+1}\right) + \rho^k A^\top \left(Ax^{k,r+1} + B\bar{x}^{k,r} + z^{k,r} + y^{k,r}/\rho^k\right) \\ &\quad + \sum_{c=1}^{C_\phi} \mu_c \nabla \phi_c\left(x^{k,r+1}\right) + \sum_{c=1}^{C_\psi} \nu_c \nabla \psi_c\left(x^{k,r+1}\right) \\ 0 &= \mu_c \phi_c\left(x^{k,r+1}\right), c = 1, \ldots, C_\phi, \quad 0 = \psi_c\left(x^{k,r+1}\right), c = 1, \ldots, C_\psi. \end{aligned} \tag{27}$$

Line 9 of Algorithm 1 is thus to solve the above equations for $x^{k,r+1}$. This can be achieved through an interior point algorithm, which employs double-layer iterations to find the solution. In the outer iteration, a barrier technique is used to convert the inequality constraints into an additional term in the objective; the optima (or stationary points) of the resulting barrier problems converge to true optima (stationary points) as the barrier parameter converges to 0. In the inner iteration, a proper search method is used to obtain the optimum of the barrier problem.Since both the interior point algorithm and the ELL algorithm 1 have a double-layer structure, using an interior point algorithm as the NLP solver in ELL results in four layers of iterations. For simplification, we consider *matching the outer layers and inner layers of the NLP algorithm and ELL, respectively. Specifically, this implies the following two modifications of the ELL algorithm.*

To incorporate the outer iterations of interior point optimization into the outer iterations of ELL, the function $f(x)$ is appended with a barrier term $-b^k \sum_{c=1}^{C_\phi} \ln\left(-\phi_c(x)\right)$, where the b^k is the barrier parameter varying with the outer iteration k and decaying to 0 as k increases. Hence a "barrier augmented Lagrangian" can be defined as

$$L_b = L - b \sum_{c=1}^{C_\phi} \ln\left(-\phi_c(x)\right). \tag{28}$$

Based on the arguments in the previous subsection, if the x-optimization step returns a $x^{k,r+1}$ minimizing L_b with respect to x, then the inner iterations result in the descent of L_{b^k}, which implies the satisfaction of conditions (23), with f modified by the barrier function. Obviously, if Assumption 3 holds for L, then it also holds for L_{b^k} when $\mathcal{X}$ has a nonempty interior. It follows that the outer iterations can find

an approximate stationary point of the original problem with the decay of barrier parameters b^k.

With the inequality constraints handled by the barrier term, the inner ADMM iterations only deals with equality-constrained NLP problems. To merge the iterative routine to solve equality-constrained NLP problems (Wächter and Biegler 2005) into the ADMM iterations, we allow to perform only *a proper amount of searching steps* instead of the entire equality-constrained NLP in each inner iteration, so that the solution to the equality-constrained NLP problem can be approached asymptotically throughout the inner iterations. For this purpose, we assume that we have at hand a solver that can find an approximate solution of the equality-constrained NLP with any prespecified tolerances of violations to the KKT conditions.

Assumption 6 Assume that for any equality-constrained smooth NLP problem

$$\min_x \chi(x) \quad \text{s.t.} \quad \psi(x) = 0 \tag{29}$$

a solver that guarantees the convergence to any approximate stationary point of the above problem with a lower objective function is available. That is, starting from any initial point x^0, for any tolerances $\epsilon_4, \epsilon_5 > 0$, within a finite number of searches the solver finds a point (x, ν) satisfying

$$d_4 = \nabla\chi(x) + \sum_{c=1}^{C_\psi} \nu_c \nabla\psi_c(x), \quad d_{5c} = \psi_c(x), c = 1, \ldots, C_\psi. \tag{30}$$

for some $\|d_4\| \le \epsilon_4$, $\|d_5\| \le \epsilon_5$, and $f(x) \le f\left(x^0\right)$. Such an approximate solution is denoted as $F\left(x^0; \chi, \psi, \epsilon_4, \epsilon_5\right)$.

The above approximate NLP solution oracle is realizable by NLP solvers where the tolerances of the KKT conditions are allowed to be specified by the user, e.g., the IPOPT solver (Wächter and Biegler 2006). Under Assumption 6, the x-update step on Line 9 of Algorithm 1 is replaced by an approximate NLP solution

$$x^{k,r+1} = F\left(x^{k,r}; \chi^{k,r}, \psi, \epsilon_4^{k,r}, \epsilon_5^{k,r}\right), \tag{31}$$

where the objective function in the current iteration is the part of barrier augmented Lagrangian L_{b^k} that is related to x with the indicator function $\mathbb{I}_{\mathcal{X}}(x)$ excluded:

$$\chi^{k,r}(x) = f(x) - b_k \sum_{c=1}^{C_\phi} \ln\left(-\phi_c(x)\right) + \frac{\rho^k}{2}\left\|Ax + B\bar{x}^{k,r} + z^{k,r} + \frac{y^{k,r}}{\rho^k}\right\|^2. \tag{32}$$

This approximate algorithm with inexact x-minimization is summarized as Algorithm 2. The inner iterations are performed until $\epsilon_4^{k,r}$ and $\epsilon_5^{k,r}$ are lower than ϵ_4^k and ϵ_5^k, respectively, and (22) holds. The outer iterations are terminated when $\epsilon_4^k \le \epsilon_4$, $\epsilon_5^k \le \epsilon_5$, the barrier parameter is sufficiently small $b^k \le \epsilon_6$, and (25) holds.

Set: *Bound of dual variables* $\left[\underline{\lambda}, \overline{\lambda}\right]$, *shrinking ratio of slack variables* $\omega \in [0, 1)$, *amplifying ratio of penalty parameter* $\gamma > 1$, *diminishing outer iteration tolerances* $\left\{\epsilon_{1,2,3,4,5,6}^k\right\}_{k=1}^{\infty} \downarrow 0$, *diminishing barrier parameters* $\left\{b^k\right\}_{k=1}^{\infty} \downarrow 0$, *terminating tolerances* $\epsilon_{1,2,3,4,5,6} > 0$;
Initialization: *Starting points* x^0, $\bar{x}^0$, z^0, *duals and bounds* $\lambda^1 \in \left[\underline{\lambda}, \overline{\lambda}\right]$, *penalty* $\beta^1 > 0$;
Outer iteration count $k \leftarrow 0$;
while $\epsilon_4^k \geq \epsilon_4$ *or* $\epsilon_5^k \geq \epsilon_5$ *or* $b^k \geq \epsilon_6$ *or stationarity criterion* (25) *is not met* **do**
  Set: *Diminishing tolerances* $\left\{\epsilon_4^{k,r}, \epsilon_5^{k,r}\right\}_{r=1}^{\infty} \downarrow 0$;
  Let $\rho^k = 2\beta^k$;
  Inner iteration count $r \leftarrow 0$;
  Initialization: $x^{k,0}$, $\bar{x}^{k,0}$, $z^{k,0}$, $y^{k,0}$ *satisfying* $\lambda^k + \beta^k z^{k,0} + y^{k,0} = 0$;
  while $\epsilon_4^{k,r} \geq \epsilon_4^k$ *or* $\epsilon_5^{k,r} \geq \epsilon_5^k$ *or stopping criterion* (22) *is not met* **do**
    $x^{k,r+1} = F\left(x^{k,r}; \chi^{k,r}, \psi, \epsilon_4^{k,r}, \epsilon_5^{k,r}\right)$, where $\chi^{k,r}$ is given by (32);
    $\bar{x}^{r+1} = G\left(B, Ax^{r+1} + z^{k,r} + y^{k,r}/\rho^k, \rho^k\right)$, where G is given by (26);
    $z^{k,r+1} = -\frac{\rho^k}{\rho^k+\beta^k}\left(Ax^{k,r+1} + B\bar{x}^{k,r+1} + \frac{y^{k,r}}{\rho^k}\right) - \frac{1}{\rho^k+\beta^k}\lambda^k$;
    $y^{k,r+1} = y^{k,r} + \rho^k\left(Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1}\right)$;
    $r \leftarrow r + 1$;
  end
  $\left(x^{k+1}, \bar{x}^{k+1}, z^{k+1}, y^{k+1}\right) \leftarrow \left(x^{k,r}, \bar{x}^{k,r}, z^{k,r}, y^{k,r}\right)$;
  Update λ^{k+1} and β^{k+1} by (19);
end

Lemma 4 (Convergence of the approximate algorithm) *Suppose that Assumptions 1–6 hold. For any outer iteration k, given any positive tolerances* $\left\{\epsilon_1^k, \ldots, \epsilon_5^k\right\}$, *within a finite number of inner iterations r, the obtained solution satisfies*

$$\begin{aligned} d_1^k + d_4^k &= \nabla f\left(x^{k,r+1}\right) + \sum_{c=1}^{C_\phi} \mu_c^{k,r+1} \nabla \phi_c\left(x^{k,r+1}\right) \\ &\quad + \sum_{c=1}^{C_\psi} \nu_c^{k,r+1} \nabla \psi_c\left(x^{k,r+1}\right) + A^\top y^{k,r+1} \\ d_2^k &\in \partial g\left(\bar{x}^{k,r+1}\right) + \mathcal{N}_{\bar{\mathcal{X}}}\left(\bar{x}^{k,r+1}\right) + B^\top y^{k,r+1}, \\ 0 &= \lambda^k + \beta^k z^{k,r+1} + y^{k,r+1} \\ d_3^k &= Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1}, \quad d_5^k = \psi\left(x^{k,r+1}\right), \\ -b^k &= \mu_c^{k,r+1} \phi_c\left(x^{k,r+1}\right), c = 1, \ldots, C_\phi. \end{aligned} \tag{33}$$

for some $d_1^k, \ldots, d_5^k$ *with* $\left\|d_1^k\right\| \leq \epsilon_1^k, \ldots, \left\|d_5^k\right\| \leq \epsilon_5^k$. *Then, suppose that the outer iteration tolerances* $\left\{\epsilon_1^k, \ldots, \epsilon_5^k\right\}$ *and barrier parameters* b^k *are diminishing with increasing k, given any terminating tolerances* $\epsilon_1, \ldots, \epsilon_6 > 0$, *within a finite number*

of outer iterations, Algorithm 2 finds a point $\left(x^{k+1}, \bar{x}^{k+1}, z^{k+1}, y^{k+1}, \mu^{k+1}, \nu^{k+1}\right)$ *satisfying*

$$\begin{aligned} d_1 + d_4 &= \nabla f\left(x^{k+1}\right) + \sum_{c=1}^{C_\phi} \mu_c^{k+1} \nabla \phi_c\left(x^{k+1}\right) + \sum_{c=1}^{C_\psi} \nu_c^{k+1} \nabla \psi_c\left(x^{k+1}\right) + A^\top y^{k+1} \\ d_2 &\in \partial g\left(\bar{x}^{k+1}\right) + \mathcal{N}_{\bar{\mathcal{X}}}\left(\bar{x}^{k+1}\right) + B^\top y^{k+1}, \quad 0 = \lambda^k + \beta^k z^{k+1} + y^{k+1} \\ d_3 &= Ax^{k+1} + B\bar{x}^{k+1}, \quad d_5 = \psi\left(x^{k+1}\right), \\ -d_6 &= \mu_c^{k+1} \phi_c\left(x^{k+1}\right), c = 1, \ldots, C_\phi. \end{aligned} \tag{34}$$

for some $d_1, \ldots, d_6$ *with* $\left\|d_j\right\| \leq \epsilon_j$, $j = 1, \ldots, 5$, $d_6 \in \left(0, \epsilon_6\right]$.

3.3 Accelerated algorithm

In the ELLA algorithm, let us make the following assumption regarding our choice of tolerances $\epsilon_4^{k,r}$ and $\epsilon_5^{k,r}$.

Assumption 7 Suppose that we choose a continuous and strictly monotonically increasing function $\pi\colon [0, \infty) \to [0, \infty)$ with $\pi(0) = 0$ such that $\epsilon_5^{k,r} = \pi\left(\epsilon_4^{k,r}\right)$, and choose $\epsilon_4^{k,r+1}$ with an analogous function of $\left\|\rho^k A^\top \left(B\bar{x}^{k,r+1} - B\bar{x}^{k,r} + z^{k,r+1} - z^{k,r}\right)\right\|$ when the resulting value is strictly smaller than the previous tolerance $\epsilon_4^{k,r}$ but not smaller the ultimate one ϵ_4^k.

The choice of function π to relate the stationarity tolerance and equality tolerance in NLP subroutine is aimed at balancing the effort to reduce both errors. The choice of $\epsilon_4^{k,r+1}$ is based on the rationale that after the rth inner iteration, the obtained solution $x^{k,r+1}$ satisfies the stationarity condition within a tolerance of $\epsilon_4^{k,r}$, and after the update of $\bar{x}$, z and y variables, the violation to the stationarity condition is bounded by $\epsilon_4^{k,r} + \left\|\rho^k A^\top (B\bar{x}^{k,r+1} - B\bar{x}^{k,r} + z^{k,r+1} - z^{k,r})\right\|$. Therefore, $\epsilon_4^{k,r}$ should be balanced with the second term, which, however, is realizable only after the $\bar{x}$- and z-updates after the x-update and hence assigned to $\epsilon_4^{k,r+1}$.

We note from Algorithm 2 that under Assumption 7, each inner iteration r is a mapping from $\left(x^{k,r}, \bar{x}^{k,r}, z^{k,r+1}, y^{k,r+1}, \epsilon_4^{k,r}\right)$ to $\left(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r+1}, y^{k,r+1}, \epsilon_4^{k,r+1}\right)$. In fact, despite the dependence of the latter variables on $x^{k,r}$ and $\epsilon_4^{k,r}$, such dependence can be ignored in the sense that the descent of the barrier augmented Lagrangian L_{β^k} will always guide the sequence of intermediate solutions towards the set of ϵ_4^k -approximate stationary points of the relaxed barrier problem. It follows that under the approximate algorithm, the sequence $\left\{\left(\bar{x}^{k,r}, z^{k,r}\right)\right\}_{r=1}^{\infty}$ will converge to a fixed point, and the convergence of $\left\{y^{k,r}\right\}$ accompanies the convergence of $\left\{z^{k,r}\right\}$. It is thus clear that we may resort to Anderson acceleration introduced in Sect. 2.3 by denoting $w = (\bar{x}, z)$, the iteration as a mapping h_0, and $h(w) = w - h_0(w)$, and collecting at the rth inner iteration the following multi-secant information about the previous m inner iterations:

$$\Delta_w^{k,r} = \left[\delta_w^{k,r-m} \ \cdots \ \delta_w^{k,r-1}\right], \Delta_h^{k,r} = \left[\delta_h^{k,r-m} \ \cdots \ \delta_h^{k,r-1}\right], \tag{35}$$

where $\delta_h^{r-m'} = w^{k,r-m'+1} - w^{k,r-m'}$ and $\delta_h^{r-m'} = h\left((w^{k,r-m'+1}\right) - h\left(w^{k,r-m'}\right)$, $m' = m-1, \ldots, 0$.

However, the possibility that Δ_w^k may not be of full rank and $H_{k,r}$ may be singular requires certain modifications to the original acceleration scheme. The following regulation technique was used in Zhang et al. (2018). To ensure the invertibility of $H_{k,r}$, the δ_h vector is perturbed to

$$\tilde{\delta}_h^{k,r-m+m'} = \left(1 - \theta_{k,r}^{m'}\right)\delta_h^{k,r-m+m'} + \theta_{k,r}^{m'}\delta_w^{k,r-m+m'}, \tag{36}$$

where the perturbation magnitude $\theta_{k,r}^{m'}$ is determined by

$$\theta_{k,r}^{m'} = \varphi\left(\frac{\left(\hat{\delta}_w^{k,r-m+m'}\right)^\top \left(H_{k,r}^{m'}\right)^{-1} \delta_h^{k,r-m+m'}}{\left\|\hat{\delta}_w^{k,r-m+m'}\right\|^2}; \eta_\theta\right). \tag{37}$$

with regularization hyperparameter $\eta_\theta \in (0,1)$. The function $\varphi(\theta;\eta)$ is defined by

$$\varphi(\theta;\eta) = \begin{cases} (\eta \text{sign}\theta - \theta)/(1-\theta), & |\theta| \le \eta \\ 0, & |\theta| > \eta \end{cases}. \tag{38}$$

With regulation, $H_{k,r}^{-1}$ is induced from $\left(H_{k,r}^0\right)^{-1} = I$ according to

$$\begin{aligned} \left(H_{k,r}^{m'+1}\right)^{-1} &= \left(H_{k,r}^{m'}\right)^{-1} \\ &+ \frac{\left(\delta_w^{k,r-m+m'} - \left(H_{k,r}^{m'}\right)^{-1}\tilde{\delta}_h^{k,r-m+m'}\right)\left(\hat{\delta}_w^{k,r-m+m'}\right)^\top \left(H_{k,r}^{m'}\right)^{-1}}{\left(\hat{\delta}_w^{k,r-m+m'}\right)^\top \left(H_{k,r}^{m'}\right)^{-1}\tilde{\delta}_h^{k,r-m+m'}} \end{aligned} \tag{39}$$

for $m' = 0, \ldots, m-1$ with $H_{k,r}^m = H_{k,r}$. To avoid the rank deficiency Δ_w, a restart checking strategy is used, where the memory is cleared when the Gram-Schmidt orthogonalization becomes ill conditioned ($\left\|\hat{\delta}_w^{k,r}\right\| < \eta_w \left\|\delta_w^{k,r}\right\|$ for some $\eta_w \in (0,1)$) or the memory exceeds a maximum M; otherwise the memory is allowed to grow. Hence the Anderson acceleration is well-conditioned.

Lemma 5 (Well-conditioning of Anderson acceleration, Zhang et al. (2018)) *Using the regularization and restart checking techniques, it is guaranteed that*

$$\left\|H_{k,r}^{-1}\right\|_2 \le \theta^{-M}\left[3\left(1+\theta+\eta_w\right)^M \eta_w^{-N} - 2\right]^{N-1} < +\infty \tag{40}$$

where M is the maximum number of steps in the memory and N is the dimension of w.

A well-conditioned Anderson acceleration is not yet sufficient to guarantee the convergence. Hence we employ a safeguarding technique modified from Zhang et al. (2018) which aims at suppressing a too large increase in the barrier augmented Lagrangian by rejecting such acceleration steps. When Anderson acceleration suggests an update from $w = (\bar{x}, z)$ to $\tilde{w} = \left(\tilde{\bar{x}}, \tilde{z}\right)$ under the current value of Ax, the resulting Lagrangian increase black, if positive, must not exceed an upper bound:

$$\Delta L_{\max} = \tilde{L}_0 \frac{\eta_L}{\left(R_+ + 1\right)^{1+\sigma}} = \left(\beta^k \left\| B\tilde{\bar{x}}^{k,1} - B\bar{x}^{k,0} \right\|^2 + \frac{\beta^k}{2} \left\| z^{k,1} - z^{k,0} \right\|^2\right) \frac{\eta_L}{\left(R_+ + 1\right)^{1+\sigma}} \tag{41}$$

where $\tilde{L}_0$ is the expected Lagrangian decrease after the first non-accelerated iteration after initialization according to Lemma 1, used as a scale for the change in the barrier augmented Lagrangian, $\eta_L, \sigma > 0$ are hyperparameters, and R_+ is the number of already accepted acceleration steps. With safeguarding, it can be guaranteed that the barrier augmented Lagrangian always stays bounded, since $\sum_{R_+=0}^{\infty} \left(R_+ + 1\right)^{-(1+\sigma)} < +\infty$. We also require that the acceleration should not lead to a drastic change in w:

$$\|\Delta w\|_{\max}^2 = \frac{\tilde{L}_0}{\beta^k} \frac{\eta_{\tilde{w}}}{\sqrt{1 + R_+}}, \tag{42}$$

where $\eta_{\tilde{w}} > 0$ is a hyperparameter. $1/\sqrt{1 + R_+}$ reflects an expected change according to the plain ADMM iteration, which is used to suppress disproportionate large deviations due to Anderson acceleration.

Finally, the accelerated algorithm using the Anderson acceleration technique for fixed-point iteration of $(\bar{x}, z)$ is summarized as Algorithm 3. This is our final ELLADA algorithm, whose convergence can now be guaranteed by the following lemma, the proof of which is given in "Appendix 4".

Lemma 6 (Convergence under Anderson acceleration) *Suppose that Assumptions 1–7 hold. Under regulated and safe-guarded Anderson acceleration, Algorithm 3 finds within a finite number of inner iterations r a point satisfying* (33). *The convergence of outer iterations to an approximate stationary point satisfying* (34) *hence follows*.

Summarizing the conclusions of all the previous lemmas in this section, we have arrived at the following theorem.

Theorem 1 *Suppose that the following assumptions hold*:

1. *Function f is lower bounded on* $\mathcal{X}$;
2. *Function g is convex and lower bounded on* $\bar{\mathcal{X}}$;

3. *Initialization of outer iterations allows a uniform upper bound of the augmented Lagrangian, e.g., a feasible solution is known a priori*;
4. *Minimization of* $g(\bar{x}) + \frac{\rho}{2}\|B\bar{x} + v\|^2$ *with respect to* $\bar{x}$ *allows an oracle* $G(B, v, \rho)$ *returning a unique solution for any* v *of appropriate dimension and* $\rho > 0$;
5. *Functions* f, ϕ, *and* ψ *are continuously differentiable, and the constraints* (ϕ, ψ) *are strictly feasible*;
6. *There exists a solver for equality-constrained NLP to any specified tolerances of KKT conditions*.

Then given any tolerances $\epsilon_1, \ldots, \epsilon_6 > 0$, *the ELLADA algorithm* (*Algorithm* 3) *gives an* $(\epsilon_1, \ldots, \epsilon_6)$*-approximate KKT point satisfying the conditions* (34).

Set: *Dual bounds* $\left[\underline{\lambda}, \overline{\lambda}\right]$, *outer iteration parameters* $\omega \in [0, 1)$, $\gamma > 1$, $\left\{\epsilon_{1,2,3,4,6}^k\right\}_{k=1}^{\infty} \downarrow 0$, $\left\{b^k\right\}_{k=1}^{\infty} \downarrow 0$, *final tolerances* $\epsilon_{1,2,3,4,6} > 0$, *function* π, *acceleration parameters* $\theta \in (0, 1)$, $\sigma > 0$, $\eta_\epsilon > 0$, $\eta_w \in (0, 1)$, $\eta_L > 0$, $\eta_{\tilde{w}} > 0$, $M \in \mathbb{N}$. Let $\epsilon_5 = \pi(\epsilon_4)$;

Initialization: *Starting points* x^0, $\bar{x}^0$, z^0, $\lambda^1 \in \left[\underline{\lambda}, \overline{\lambda}\right]$, *penalty parameter* $\beta^1 > 0$, $\epsilon_5^0 = \pi\left(\epsilon_4^0\right)$;

Outer iteration count $k \leftarrow 0$;

while $\epsilon_4^k \geq \epsilon_4$ *or* $\epsilon_5^k \geq \epsilon_5$ *or* $b^k \geq \epsilon_6$ *or stationarity criterion* (25) *is not met* **do**

  Set: *Initial tolerances* $\epsilon_4^{k,0}$, $\epsilon_5^{k,0} = \pi(\epsilon_4^{k,0})$, *penalty* $\rho^k = 2\beta^k$, *Jacobian estimate* $H_{k,0}^{-1} = I$;

  Inner iteration count $r \leftarrow 0$, count of accelerated steps $R_+^k = 0$, memory length $m \leftarrow 0$;

  Initialization: $x^{k,0}$, $\bar{x}^{k,0}$, $z^{k,0}$, $y^{k,0}$ *satisfying* $\lambda^k + \beta^k z^{k,0} + y^{k,0} = 0$;

  while $\epsilon_4^{k,r} \geq \epsilon_4^k$ *or* $\epsilon_5^{k,r} \geq \epsilon_5^k$ *or stopping criterion* (22) *is not met* **do**

    $x^{k,r+1} = F\left(x^{k,r}; \chi^{k,r}, \psi, \epsilon_4^{k,r}, \epsilon_5^{k,r}\right)$, where $\chi^{k,r}$ is given by (32);

    $\bar{x}^{k,r+1} = G\left(B, Ax^{r+1} + z^{k,r} + y^{k,r}/\rho^k, \rho^k\right)$, where G is given by (26);

    $z^{k,r+1} = -\frac{\rho^k}{\rho^k+\beta^k}\left(Ax^{k,r+1} + B\bar{x}^{k,r+1} + \frac{y^{k,r}}{\rho^k}\right) - \frac{1}{\rho^k+\beta^k}\lambda^k$;

    $y^{k,r+1} = y^{k,r} + \rho^k\left(Ax^{k,r+1} + B\bar{x}^{k,r+1} + z^{k,r+1}\right)$;

    $\tilde{y}^{k,r} = -\lambda^k - \beta^k \tilde{z}^{k,r}$;

    $\tilde{x}^{k,r+1} = F\left(x^{k,r}; \tilde{\chi}^{k,r}, \psi, \epsilon_4^{k,r}, \epsilon_5^{k,r}\right)$, with $\tilde{\chi}$ in (32) with $\bar{x}, z, y$ replaced by $\tilde{\bar{x}}, \tilde{z}, \tilde{y}$;

    $\tilde{\bar{x}}^{k,r+1} = G\left(B, A\tilde{x}^{r+1} + \tilde{z}^{k,r} + \tilde{y}^{k,r}/\rho^k, \rho^k\right)$;

    $\tilde{z}^{k,r+1} = -\frac{\rho^k}{\rho^k+\beta^k}\left(A\tilde{x}^{k,r+1} + B\tilde{\bar{x}}^{k,r+1} + \frac{\tilde{y}^{k,r}}{\rho^k}\right) - \frac{1}{\rho^k+\beta^k}\lambda^k$;

    if $r = 0$ **then**

      $\tilde{w}^{k,1} \leftarrow \left(\bar{x}^{k,1}, z^{k,1}\right)$, and calculate $\tilde{L}_0$ by (41);

    else

      $\delta_w^{k,r-1} = \tilde{w}^{k,r} - w^{k,r-1}$, $\delta_h^{k,r} = \tilde{w}^{k,r} - \tilde{w}^{k,r+1} - w^{k,r-1} + w^{k,r}$, $m \leftarrow m + 1$;

      $\hat{\delta}_w^{k,r-1} = \delta_w^{k,r-1} - \sum_{m'=2}^{m} \frac{\left(\hat{\delta}_w^{k,r-m'}\right)^\top \delta_w^{k,r-1}}{\left\|\hat{\delta}_w^{k,r-m'}\right\|^2} \hat{\delta}_w^{k,r-m'}$;

      if $m = M + 1$ *or* $\frac{\left\|\hat{\delta}_w^{k,r-1}\right\|}{\left\|\delta_w^{k,r-1}\right\|} < \eta_w$ **then** $m \leftarrow 0$, $\hat{\delta}_w^{k,r-1} \leftarrow \delta_w^{k,r-1}$, and $H_{k,r-1}^{-1} \leftarrow I$;

      Obtain $\tilde{\delta}_h^{k,r-1}$ by (36) with $m' = m - 1$ and $\theta_{k,r-1} = \varphi\left(\frac{\left(\hat{\delta}_w^{k,r-1}\right)^\top H_{k,r-1}^{-1} \delta_h^{k,r-1}}{\left\|\hat{\delta}_w^{k,r-1}\right\|^2}; \theta\right)$;

      Update $H_{k,r}^{-1} = H_{k,r-1}^{-1} + \frac{\left(\delta_w^{k,r-1} - H_{k,r-1}^{-1}\delta_h^{k,r-1}\right)\left(\hat{\delta}_w^{k,r-1}\right)^\top H_{k,r-1}^{-1}}{\left(\hat{\delta}_w^{k,r-1}\right)^\top H_{k,r-1}^{-1}\tilde{\delta}_w^{k,r-1}}$, and suggest $\tilde{w}^{k,r+1} = w^{k,r} - H_{k,r}^{-1}\left(w^{k,r} - w^{k,r+1}\right)$;

      if *criteria* (41) *and* (42) *hold* **then** $w^{k,r+1} \leftarrow \tilde{w}^{k,r+1}$, $y^{k,r+1} \leftarrow -\lambda^k - \beta^k \tilde{z}^{k,r+1}$;

    end

    $\epsilon_4^{k,r+1} = \left\|\rho^k A^\top\left(B\tilde{\bar{x}}^{k,r+1} - B\bar{x}^{k,r} + z^{k,r+1} - z^{k,r}\right)\right\|$, $\epsilon_5^{k,r+1} = \pi\left(\epsilon_4^{k,r+1}\right)$;

    $r \leftarrow r + 1$;

  end

  $\left(x^{k+1}, \bar{x}^{k+1}, z^{k+1}, y^{k+1}\right) \leftarrow \left(x^{k,r}, \bar{x}^{k,r}, z^{k,r}, y^{k,r}\right)$;

  Update λ^{k+1} and β^{k+1} by (19);

  $k \leftarrow k + 1$;

end

3.4 Parameter tuning

We have so far proved the theoretical convergence of the proposed ELLADA algorithm (Algorithm 3). It is not surprising to expect that such an algorithm that combines multiple techniques aiming at improving the computational performance will require some parameter tuning effort, although these parameters do not alter the ultimate convergence. Three main groups of parameters are involved here.

The first group of parameters is related to the outer iterations, namely the threshold of slack variable decay ω and the triggered penalty amplifying ratio γ. Although the bounds of the dual variables $\left[\underline{\lambda}, \overline{\lambda}\right]$ also appear as parameters, they are expected to play a theoretical role in establishing convergence rather than practically affecting the numerical performance. Generally, smaller ω and larger γ values result in more radical increase of the penalty parameter, which on one hand reduces the number of outer iterations needed for the slack variables to converge to zero, but on the other hand, is prone the making the problem ill-conditioned and increasing the number of inner iterations. Hence, ω and γ should be at neither too large nor too small values, which can be tuned by fixing the tolerance parameters at some conservative levels (to be improved by the next step), and examining when the ELL algorithm is the most efficient.

The second groups of parameters are the tolerances associated with the approximate NLP solution and inner layer ADMM iterations. Since $B\bar{x}^{k,r+1} - B\bar{x}^{k,r}$ and $z^{k,r+1} - z^{k,r}$ are related to the first two terms in the linear constraint $Ax + B\bar{x} + z = 0$ and hence of similar magnitudes, ϵ_1 and ϵ_2 as well as ϵ_1^k and ϵ_2^k during the outer iterations k should be similar. According to Assumption 7, $\epsilon_4^{k,r}$ should be chosen accordingly with $\epsilon_1^{k,r}$, and ϵ_4^k accordingly with ϵ_1. Across the outer iterations, these tolerances are required to be exponentially decaying so that only a few outer iterations should be needed. To obtain a suitable relation π between stationarity tolerances $\epsilon_4^{k,r}$, ϵ_4^k and equality constraint tolerances $\epsilon_5^{k,r}$, ϵ_5^k of the NLP solver, we simply let π to be a linear function. That is,

$$\begin{aligned}
&\epsilon_1 = \epsilon_2 = \epsilon_4,\\
&\epsilon_1^k = \epsilon_2^k = \epsilon_4^k = \max\left(c_1/a_1^{k-1}, \epsilon_1\right), \epsilon_5^k = \frac{\epsilon_5}{\epsilon_4}\epsilon_1^k,\\
&\epsilon_4^{k,r} = \max\left(c_4\left(\epsilon_1^{k,r}\right)^{a_4}, \epsilon_4^k\right), \epsilon_5^{k,r} = \frac{\epsilon_5}{\epsilon_4}\epsilon_4^{k,r}.
\end{aligned} \tag{43}$$

The primal and dual residuals ϵ_1 and ϵ_3 should also be set in proportion so that the obtained solution is an approximate stationary point rather than a stagnant point. For the barrier constant, we need to guarantee that its diminishing should be in pace with the outer iterations, and hence can be set based on ϵ_3^k. We also require that the barrier constants should be clamped within a specified range. That is,

$$\epsilon_3^k = \frac{\epsilon_3}{\epsilon_1}\epsilon_1^k, b^k = \min\left(b^{\max}, \max\left(\epsilon_6, \epsilon_6\left(\epsilon_1^k/\epsilon_1\right)^{a_6}\right)\right). \tag{44}$$

All the tolerance parameters thus reduce to c_1, a_1, c_4, a_4, a_6 and the limits, $\epsilon_1, \epsilon_3, \epsilon_4, \epsilon_5, \epsilon_6$, $b^{\max}$. The limits should be determined at the values at which the solution is satisfactory while not causing excessive computational cost. The exponents a_4 and a_6 can be chosen between 1 or 2. Then there exists only three parameters c_1, a_1, c_4 to be determined empirically so that the numerical performance of the resulting ELLA algorithm best improves that of ELL.

The remaining 6 parameters are related to Anderson acceleration and its auxiliary regulation and safeguarding techniques – $(\eta_\theta, \eta_w, M, \sigma, \eta_{\tilde{w}}, \eta_L)$, which can be classified into 3 groups, namely η_θ involved in regularization, (η_w, M) in memory clean-up, and $(\sigma, \eta_{\tilde{w}}, \eta_L)$ in rejecting inappropriate acceleration candidates. Since η_θ aims at guaranteeing the invertibility of the Jacobian estimate with a perturbation, its value should be reasonably small (e.g., $\eta_\theta = 0.05$ or 0.01), and the same holds for η_w. The maximum allowed memory M should be large enough to obtain a meaningful "average" curvature, but should not be too large to include too many obsolete previous points. Empirically, nevertheless, it was found that the variation of η_θ, η_w, and M does not have a strong impact. For the last three safeguarding-related parameters, the decrease of σ or the increase of $\eta_{\tilde{w}}$ or η_L allows more accepted Anderson acceleration steps. The effects on the overall computational performance, however, are non-intuitive, since more accelerations may imply not only faster convergence but also possibly more overshoot and oscillations. Therefore, careful tuning is needed to obtain an appropriate extent of acceleration.

4 Nonlinear optimization benchmark

To test the proposed ELLADA algorithm as a generic method of nonconvex constrained distributed optimization, we consider a problem from the NLP benchmark library in this section, while the implementation and a case study on distributed nonlinear MPC will be discussed in the next sections. The benchmark is the `camshape` problem from the COPS (constrained optimization problem set) library (Dolan and Moré 2002), described as follows:

$$\begin{aligned} \min_{r_1,\dots,r_n} \quad & \sum_{i=1}^{n} r_i \\ \text{s.t.} \quad & r_{\min} \le r_i \le r_{\max}, i = 1, \dots, n \\ & 2r_{i-1}r_{i+1}\cos\theta \le r_i\left(r_{i-1} + r_{i+1}\right), i = 0, 1, \dots, n, n+1 \\ & -\alpha \le \frac{r_{i+1} - r_i}{\theta} \le \alpha, i = 0, 1, \dots, n \end{aligned} \tag{45}$$

where $r_{\min} = 1$, $r_{\max} = 2$, $\alpha = 1.5$, $\theta = 2\pi/5(n+1)$, $r_{-1} = r_0 = r_{\min}$, $r_{n+1} = r_{\max}$, $r_{n+2} = r_n$. We consider solving this problem with $n = 4n_0 + 2$, $n_0 = 100$, in a distributed manner with 4 subproblems. The variables r_{in_0+1} and r_{in_0+2}, $i = 1, 2, 3$ are the shared variables between the neighboring subproblems, and hence $\bar{x}$ are created for these variables, and equality constraints are imposed (for example, the last component of x_1 and the second component of x_2 should both be equal to the second

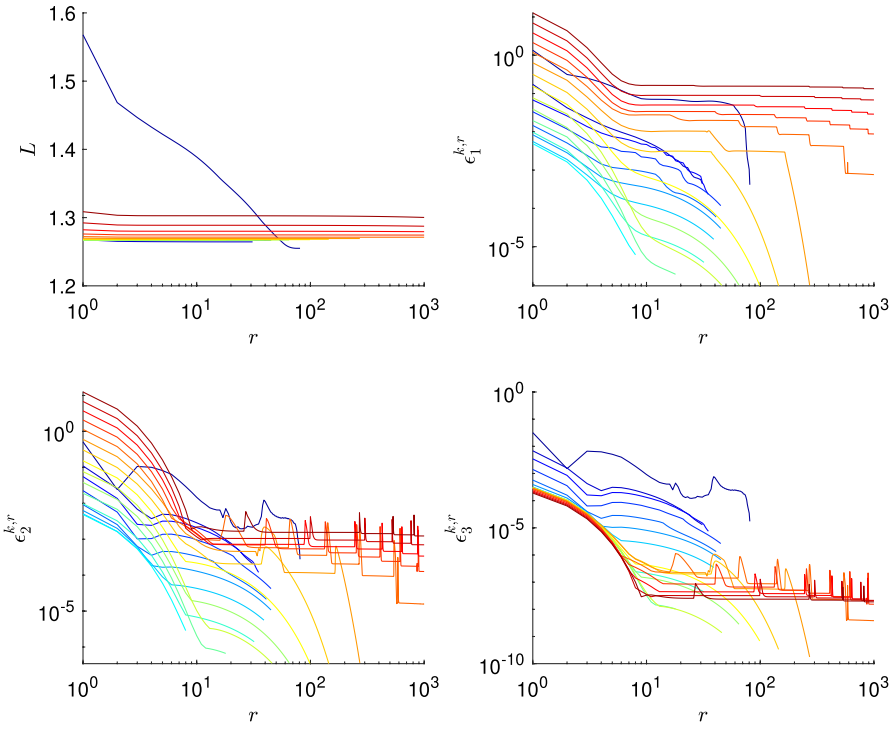

Fig. 2 Results of the ELL algorithm for the `camshape` problem. Colors from blue to red correspond to increasing outer iterations. (Color figure online)

component of $\bar{x}$). Thus, formulations in the form of (20) and subsequently of (21) are obtained, with the dimensions of $\bar{x}$ and z being 6 and 12, respectively. Each subproblem contains 102 variables, 100 or 101 nonlinear inequality constraints, and 202 linear inequality constraints. The subproblems are handled with the IPOPT solver used in OPTI Toolbox of Matlab. The gradient of the objective function, the Jacobian matrix of the nonlinear constraints and its structure are provided to the solver.

For ELL, the final tolerances $\epsilon_1 = \epsilon_2 = 10^{-6}$, $\epsilon_3 = 10^{-4}$ are set as equal to the default values in the IPOPT solver, and the parameters are tuned as $\omega = 0.75$, $\gamma = 2$, $\epsilon_1^k = \epsilon_2^k = 10^{-3}/2^{k-1}$, $\epsilon_3^k = 10^{-1}/2^{k-1}$. The initial penalty parameter $\beta = 4$. The maximum number of outer iterations is set as 20, and the maximum number of inner iterations for each outer iteration is set as 1000. The intermediate calculated results of augmented Lagrangian L and $\epsilon_1^{k,r}$, $\epsilon_2^{k,r}$, $\epsilon_3^{k,r}$ are shown in Fig. 2. The expected monotonic decreasing property of L is observed, leading to the convergence of the asymptotic inner iterations. However, for all the outer iterations after the 15th, the maximum allowed number of inner iterations is reached, and the maximum allowed number of outer iterations is also reached, resulting in a total number of 6002 inner iterations and a computational time of 761.40 s (0.1269 s for each inner iteration on average). At the end of the 20 outer iterations, $\|z\|_\infty = 5.69 \times 10^{-4}$ remains high. This is due to the large value of the penalty parameter values (above 2^{10}), which fundamentally limits the speed of convergence.

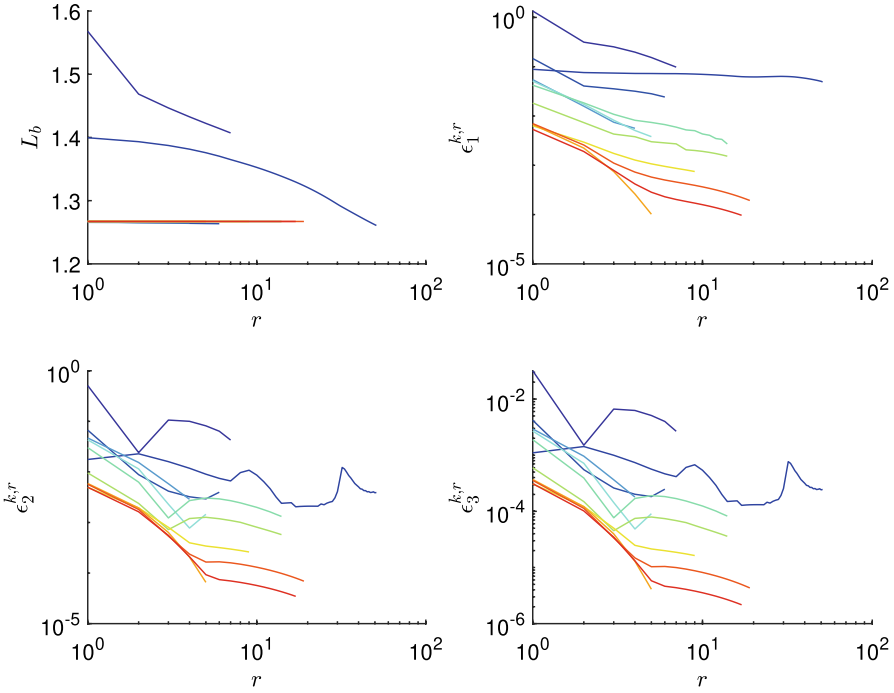

Fig. 3 Results of the ELLA algorithm for the `camshape` problem. Colors from blue to red correspond to increasing outer iterations. (Color figure online)

By observing Fig. 2, for ELLA we decide to reset the final tolerances to coarser values $\epsilon_1 = \epsilon_2 = \epsilon_4 = 10^{-4}$, $\epsilon_3 = 10^{-3}$ since it appears difficult for the solver to practically reach lower precisions. We let $\epsilon_1^{k,r} = \epsilon_2^{k,r} = 10^{-1}/2^{k-1}$, $\epsilon_5 = 10^{-4}$, $\epsilon_6 = 10^{-4}$. The barrier parameter is set as $b^k = \epsilon_6(\epsilon_3^k/\epsilon_3)^2$. The total number of inner iterations needed to converge is 151 in 11 outer iterations, with a computational time of 14.38 s. In contrast, ELL achieves these tolerances in 7 outer iterations, which takes 317 inner iterations and 33.67 s. Hence, ELLA needs 47.6% less iterations and 57.3 % less computational time than ELL to reach solutions with the same level of tolerances. On average for each inner iteration, 10.3 % of computational time is saved compared to ELL. In ELLA, the behaviors of the barrier augmented Lagrangian L_b and $\epsilon_{1,2,3}^{k,r}$, as shown in Fig. 3, are similar to that of ELL.

Finally we seek to decrease the number of inner iterations needed by using ELLADA. The Anderson acceleration-related parameters are tuned as $\eta_w = 0.001$, $\theta = 0.001$, $\sigma = 0.1$, $\eta_L = 5$, $\eta_{\tilde{w}} = 1$, and $M = 3$. After acceleration, the total number of inner iterations needed becomes 84, 44.4% lower than ELLA. Within these inner iterations, 72 iterations are under Anderson acceleration. The resulting computational time is 8.1010 s, 43.7% lower than that of ELLA. It is noted from Fig. 4 that under ELLADA, the curves of $\epsilon_{1,2,3}^{k,r}$ becomes more kinky and larger deviations from monotonicity. Such phenomena highlight the necessity of safeguarding techniques to curb the behavior of Anderson acceleration as well as careful tuning of the relevant parameters.

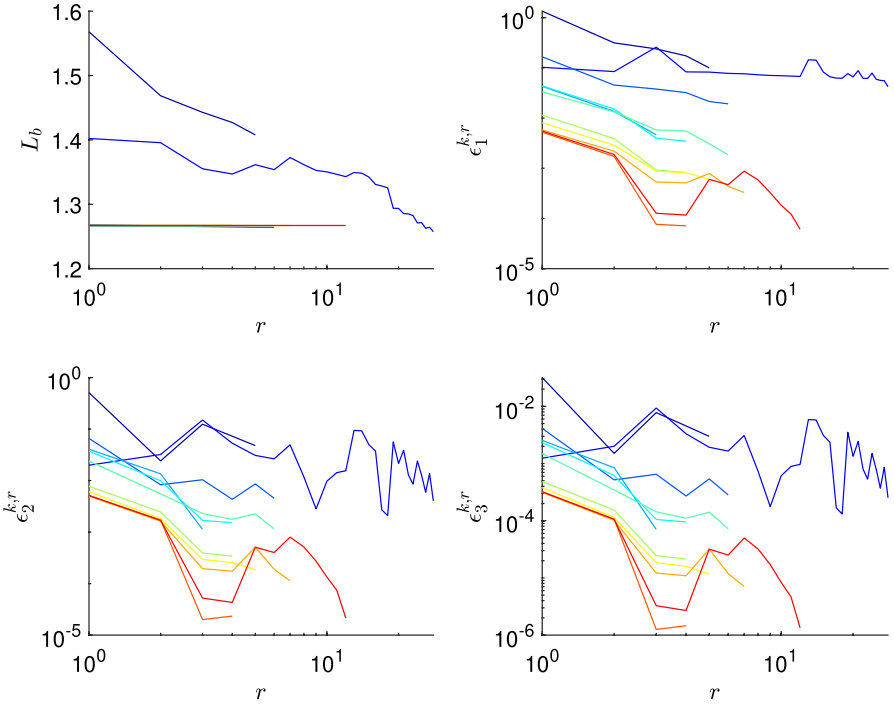

Fig. 4 Results of the ELLADA algorithm for the `camshape` problem. Colors from blue to red correspond to increasing outer iterations. (Color figure online)

5 Implementation on distributed nonlinear MPC

Consider a nonlinear discrete-time dynamical system

$$x(t+1) = f(x(t), u(t)) \tag{46}$$

where $x(t) \in \mathbb{R}^n$ and $u(t) \in \mathbb{R}^m$ are the vectors of states and inputs, respectively, for $t = 0, 1, 2, \ldots$, and $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$. Suppose that at time t we have the current states $x = x(t)$, then in MPC, the control inputs are determined by the following optimal control problem:

$$\begin{aligned} \min \quad & J = \sum_{\tau=t}^{t+T-1} \ell(\hat{x}(\tau), \hat{u}(\tau)) + \ell^{\mathrm{f}}(\hat{x}(t+T)) \\ \text{s.t.} \quad & \hat{x}(\tau+1) = f(\hat{x}(\tau), \hat{u}(\tau)), \tau = t, \ldots, t+T-1 \\ & p(\hat{x}(\tau), \hat{u}(\tau), \tau) \le 0, \tau = t, \ldots, t+T-1 \\ & q(\hat{x}(\tau), \hat{u}(\tau), \tau) = 0, \tau = t, \ldots, t+T-1 \\ & \hat{x}(t) = x. \end{aligned} \tag{47}$$

In the above formulation, the optimization variables $\hat{x}(\tau)$ and $\hat{u}(\tau)$ represent the predicted states and inputs in a future horizon $\{t, t+1, \dots, t+T\}$ with length $T \in \mathbb{N}$. The predicted trajectory is constrained by the dynamics (46) as well as some additional path constraints p, q such as the bounds on the inputs and states or Lyapunov descent to enforce stability. Functions ℓ and ℓ^{f} are called the stage cost and terminal cost, respectively. By solving (47), one executes $u(t) = \hat{u}(t)$. For simplicity it is assumed here that the states are observable; otherwise, the states can be estimated using an optimization formulation such as moving horizon estimation (MHE). For continuous-time systems, collocation techniques can be used to discretize the resulting optimal control problem into a finite-dimensional one.

Now suppose that the system (46) is large-scale with its states and outputs decomposed into n subsystems: $x = \left[x_1^\top, x_2^\top, \dots, x_n^\top\right]^\top$, $u = \left[u_1^\top, u_2^\top, \dots, u_n^\top\right]^\top$, and that the optimal control problem should be solved by the corresponding n agents, each containing the model of its own subsystem:

$$x_i(\tau+1) = f_i\Big(x_i(\tau), u_i(\tau), \left\{x_{ji}(\tau), u_{ji}(\tau)\right\}_{j\in\mathcal{P}(i)}\Big). \tag{48}$$

where $\left\{x_{ji}, u_{ji}\right\}$ stands for the states and inputs in subsystem j (i.e., components of x_j and u_j) that appear in the arguments of f_i, which comprise of the components of f corresponding to the ith subsystem. $\mathcal{P}_i$ is the collection of subsystems j that has some inputs and outputs influencing subsystem i. We assume that the cost functions and the path constraints are separable:

$$\begin{aligned} \ell(\hat{x}, \hat{u}) &= \sum_{i=1}^{n} \ell_i\left(\hat{x}_i, \hat{u}_i\right), \ell^{\mathrm{f}}(\hat{x}) = \sum_{i=1}^{n} \ell_i^{\mathrm{f}}\left(\hat{x}_i\right), \\ p(\hat{x}, \hat{u}, \tau) &= \left[p_1\left(\hat{x}_1, \hat{u}_1, \tau\right)^\top, \dots, p_n\left(\hat{x}_n, \hat{u}_n, \tau\right)^\top\right]^\top, \\ q(\hat{x}, \hat{u}, \tau) &= \left[q_1\left(\hat{x}_1, \hat{u}_1, \tau\right)^\top, \dots, q_n\left(\hat{x}_n, \hat{u}_n, \tau\right)^\top\right]^\top. \end{aligned} \tag{49}$$

5.1 Formulation on directed and bipartite graphs

Graphs or networks are systematic tools to visualize distributed control/optimization problems and allow the practitioners of distributed MPC to construct and configure their specific problem of interest in a structured manner, by using network decomposition methods to automatically generate subsystems (Daoutidis et al. 2019) and associating the subsystems and their interactions with a graph topology (Jalving et al. 2019). To better illustrate the application of ELLADA on the distributed solution of the optimal control problem (47) reformulated into the form of (21), we introduce some graph-theoretic representations of optimization problems. For problem (47), we first define a directed graph (digraph), which is a straightforward characterization of the mutual impacts among the subsystem models.

Definition 1 (*Digraph*) The digraph of system (46) under the decomposition $x = \left[x_1^\top, x_2^\top, \ldots, x_n^\top\right]^\top$ and $u = \left[u_1^\top, u_2^\top, \ldots, u_n^\top\right]^\top$ is $\mathcal{G}_1 = \left\{\mathcal{V}_1, \mathcal{E}_1\right\}$ with nodes $\mathcal{V}_1 = \{1, 2, \ldots, n\}$ and edges $\mathcal{E}_1 = \{(j,i) | j \in \mathcal{P}(i)\}$. If $(i,j) \in \mathcal{E}_1$, i.e., $j \in \mathcal{P}(i)$, we say that j is a parent of i and i is a child of j (denoted as $i \in \mathcal{C}(j)$).

Then under the decomposition, (47) can be written as

$$\begin{aligned}
\min \quad & \sum_{i\in\mathcal{V}_1} J_i = \sum_{i\in\mathcal{V}} \sum_{\tau=t}^{t+T-1} \ell_i\left(\hat{x}_i(\tau), \hat{u}_i(\tau)\right) + \ell_i^{\mathrm{f}}\left(\hat{x}_i(t+T)\right) \\
\text{s.t.} \quad & \hat{x}_i(\tau+1) = f_i\Big(\hat{x}_i(\tau), \hat{u}_i(\tau), \left\{\hat{x}_{ji}(\tau), \hat{u}_{ji}(\tau)\right\}_{j\in\mathcal{P}(i)}\Big), \\
& p_i\left(\hat{x}_i(\tau), \hat{u}_i(\tau), \tau\right) \le 0, \tau = t, \ldots, t+T-1, i \in \mathcal{V}_1 \\
& q_i\left(\hat{x}_i(\tau), \hat{u}_i(\tau), \tau\right) = 0, \tau = t, \ldots, t+T-1, i \in \mathcal{V}_1 \\
& \hat{x}_i(t) = x_i, i \in \mathcal{V}_1,
\end{aligned} \tag{50}$$

We denote the variables of the ith agent as

$$\begin{aligned}
\xi_i = \big[&\hat{x}_i(t)^\top, \hat{u}_i(t)^\top, \ldots, \hat{x}_i(t+T-1)^\top, \hat{u}_i(t+T-1)^\top, \hat{x}_i(t+T)^\top, \\
&\left\{\hat{x}_{ji}(t)^\top, \hat{u}_{ji}(t)^\top\right\}_{j\in\mathcal{P}(i)}, \ldots, \left\{\hat{x}_{ji}(t+T-1)^\top, \hat{u}_{ji}(t+T-1)^\top\right\}_{j\in\mathcal{P}(i)}\big]^\top,
\end{aligned} \tag{51}$$

in which the variables related to the jth subsystem are denoted as ξ_{ji}. Since ξ_{ji} is a part of the predicted states and inputs from subsystem j, i.e., some components of ξ_j, the interactions between the parent j and the child i be captured by a matrix $\overrightarrow{D}_{ji}$ with exactly one unit entry ("1") on every row: $\xi_{ji} = \overrightarrow{D}_{ji}\xi_j$, where the right arrow represents the impact of the parent subsystem j on the child subsystem i. By denoting the model and path constraints in agent i as $\xi_i \in \Xi_i$, the optimal control problem (46) is expressed in a compact way as follows:

$$\min \sum_{i\in\mathcal{V}_1} J_i\left(\xi_i\right) \quad \text{s.t.} \xi_i \in \Xi_i, i \in \mathcal{V}_1, \xi_{ji} = \overrightarrow{D}_{ji}\xi_j, (j,i) \in \mathcal{E}_1. \tag{52}$$

This is an optimization problem defined on a *directed graph*. An illustration for a simple case when $\mathcal{E}_1 = \{(1,2), (2,3), (3,1)\}$ is shown in Fig. 5a.

Although it is natural to represent the interactions among the subsystems in a digraph, performing distributed optimization on digraphs where the agents communicate among themselves without a coordinator can be challenging. For example, it is known that the ADMM algorithm, which behaves well for distributed optimization with 2 blocks of variables, can become divergent when directly extended to multi-block problems (Chen et al. 2016). Hence we construct such a 2-block architecture by using a *bipartite graph*.

Definition 2 (*Bipartite graph*) The bipartite graph of system (46) $\mathcal{G}_2$ is constructed from the digraph $\mathcal{G}_1$ by taking both the nodes and edges as the new nodes, and adding an edge between $i \in \mathcal{V}_1$ and $e \in \mathcal{E}_1$ if i is the

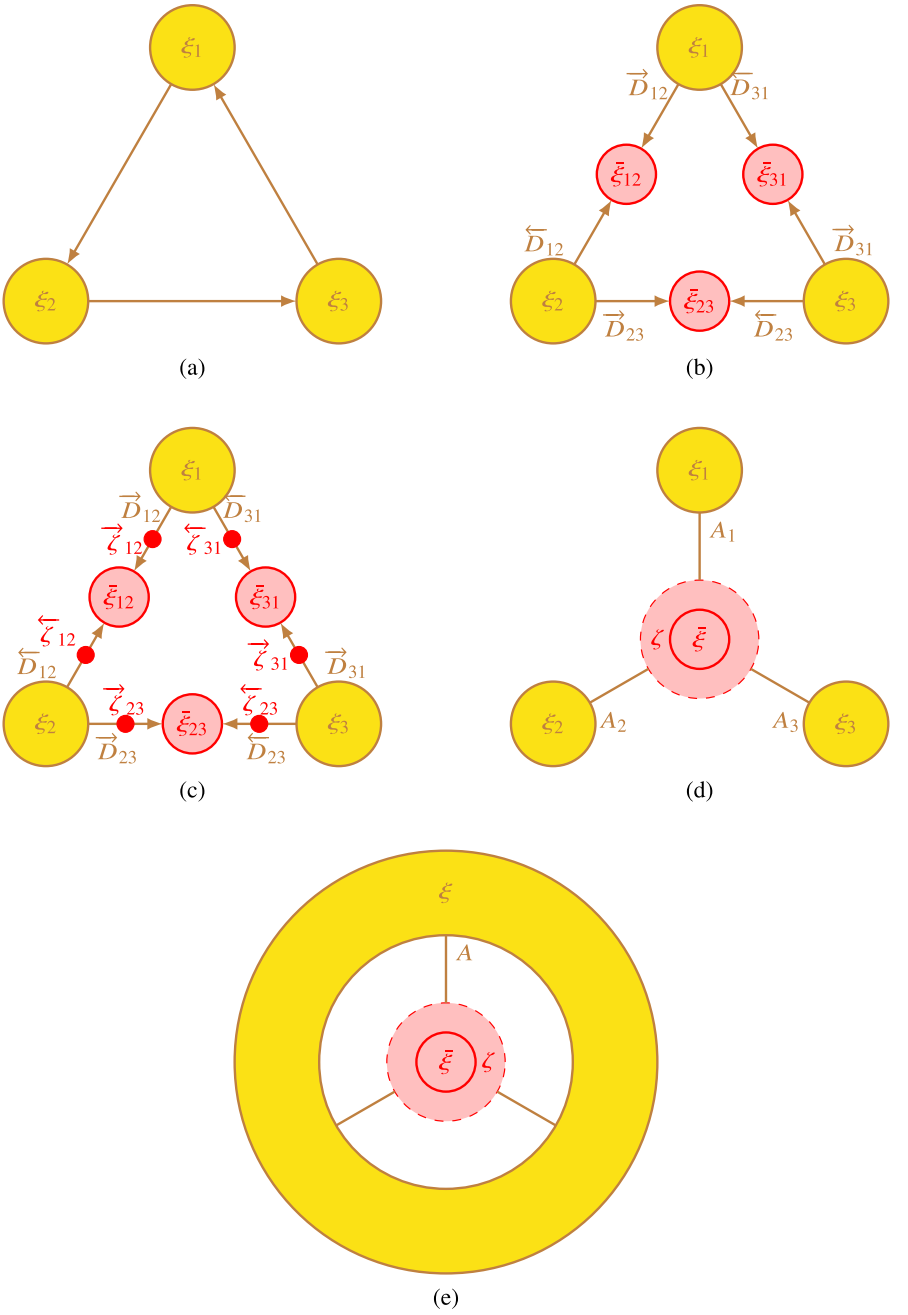

Fig. 5 Graphical illustrations of the problem structure of distributed MPC

head or tail of e in the digraph, i.e., $\mathcal{G}_2 = \left(\mathcal{V}_2, \mathcal{E}_2\right)$ with $\mathcal{V}_2 = \mathcal{V}_1 \cup \mathcal{E}_1$, $\mathcal{E}_2 = \left\{(i,e) | i \in \mathcal{V}_1, e \in \mathcal{E}_1, e = (i,j), j \in \mathcal{C}(i) \text{ or } e = (j,i), j \in \mathcal{P}(i)\right\}$.

Such a graph is bipartite since any edge is between a node of $\mathcal{V}_1$ and a node of $\mathcal{E}_1$.

We note that the last line of (52) corresponds to the digraph edges $\mathcal{E}_1$. In the bipartite graph, these edges should become nodes and hence new groups of variables should be associated with them. For this purpose, we simply need to pull out ξ_{ji} as overlapping variables $\bar{\xi}_{ji}$, and add the constraint that $\bar{\xi}_{ji}$ are some selected components of ξ_i: $\xi_{ji} = \overleftarrow{D}_{ji}\xi_i$:

$$\min \sum_{i\in\mathcal{V}_1} J_i(\xi_i) \quad \text{s.t. } \xi_i \in \Xi_i, i \in \mathcal{V}_1, \bar{\xi}_{ji} = \overrightarrow{D}_{ji}\xi_j = \overleftarrow{D}_{ji}\xi_j, (j,i) \in \mathcal{E}_1 \tag{53}$$

In (53), variables ξ_i ($i \in \mathcal{V}_1$) and $\bar{\xi}_{ji}$ ($(j,i) \in \mathcal{E}_1$) are defined on the nodes of the bipartite graph, and the constraints captured by the matrices $\overrightarrow{D}_{ji}$ and $\overleftarrow{D}_{ji}$ correspond to the bipartite edges $(j,(j,i))$ and $(i,(j,i))$, respectively. We may also write the last line of (53) as

$$\bar{\xi}_e = D_{ie}\xi_i, (i,e) \in \mathcal{E}_2. \tag{54}$$

Therefore (53) is an optimization problem on the bipartite graph. An illustration is given in Fig. 5b. Under this reformulation, the problem structure becomes a 2-block one—distributed agents $i = 1, \ldots, N$ manage the decision variables ξ_i, $\mathcal{V}_1$ in parallel without interference, and the coordinator regulates the agents by using overlapping variables $\bar{\xi}_e$, $e \in \mathcal{E}_1$.

5.2 Reformulation with slack variables

It is known that a key condition for distributed optimization in the context of the ADMM algorithm to converge is that one block of variables can always be made feasible given the other block (Wang et al. 2019). Unfortunately this condition is not always met by the problem (53). For example, given ξ_1 and ξ_2, there may not be a $\bar{\xi}_{12}$ satisfying both $\bar{\xi}_{12} = \overrightarrow{D}_{12}\xi_1$ and $\bar{\xi}_{12} = \overleftarrow{D}_{12}\xi_2$. To deal with this issue, it was proposed to associate with each linear constraint in (53), namely each edge in the bipartite graph, a slack variable ζ_{ie} (e.g., Sun and Sun 2019):

$$\begin{aligned} \min \quad & \sum_{i\in\mathcal{V}_1} J_i(\xi_i) \\ \text{s.t.} \quad & \xi_i \in \Xi_i, i \in \mathcal{V}_1 \\ & D_{ie}\xi_i - \bar{\xi}_e + \zeta_{ie} = 0, (i,e) \in \mathcal{E}_2 \\ & \zeta_{ie} = 0, (i,e) \in \mathcal{E}_2. \end{aligned} \tag{55}$$

Similar to the notation for D, we write ζ_{ie} as $\overrightarrow{\zeta}_{ij}$ if $e = (i,j)$ and $\overleftarrow{\zeta}_{ij}$ if $e = (j,i)$. Such a problem structure is graphically illustrated in Fig. 5c.

Finally, we stack all the subscripted variables into ξ, $\bar{\xi}$, ζ in a proper ordering of $i \in \mathcal{V}_1$, $e \in \mathcal{E}_1$, and $(i,e) \in \mathcal{E}_2$. The matrices D_{ie} are stacked in a block diagonal pattern in the same ordering of $(i,e) \in \mathcal{E}_2$ into A. The appearance of $\bar{\xi}_e$ in the equality constraints is represented by a matrix B (satisfying $B^\top B = 2I$). We write the

objective function as $J(\xi)$, and the set constraints Ξ_i are lumped into a Cartesian product $\Xi = \times_{i\in\mathcal{V}_1}\Xi_i$. Finally, we reach a compact formulation for (55):

$$\min J(\xi) \quad \text{s.t.}\xi \in \Xi, A\xi + B\bar{\xi} + \zeta = 0, \zeta = 0 \tag{56}$$

Such an architecture is represented by the graphs in Fig. 5d, e. The variables $\bar{\xi}$ and ζ belong to the coordinator (marked in red), and ξ is in the distributed agents. Clearly, the optimal control problem formulated as (55) with the afore-mentioned graph representations is now a special form of (21), with ξ, $\bar{\xi}$ and ζ rewritten as x, $\bar{x}$ and z, respectively, and $g(\bar{x}) = 0$, $\bar{\mathcal{X}}$ equal to the entire Euclidean space.

5.3 Implementation of ELLADA

As long as the cost function J is lower bounded (e.g., a quadratic cost), it follows from Theorem 1 that Algorithm 3 is applicable to (55), where the operations on $\bar{x}$, z, y are performed by the coordinator, and the operations on x is handled by the distributed agents. Specifically,

- The update steps of $\bar{x}, z, y$ (Lines 10–13, 15, 16) and the entire Anderson acceleration (Lines 17–26) belong to the coordinator. The updates of penalty parameters and outer-layer dual variables λ (Lines 31) should also be performed by the coordinator. The conditions for $\epsilon_1^k, \epsilon_2^k, \epsilon_3^k$ and $\epsilon_1, \epsilon_2, \epsilon_3$ are checked by the coordinator.
- The distributed agents are responsible for carrying out a trial x-update step for the Anderson acceleration (Line 9) as well as the plain x-update (Line 14). The conditions and updates for $\epsilon_4^{k,r}, \epsilon_5^{k,r}, \epsilon_4^k, \epsilon_5^k$, and $\epsilon_4, \epsilon_5, \epsilon_6$ are checked by the agents.

When executing the updates, the agents need the values of $B\bar{x} + z + y/\rho$ to add to Ax, and the coordinator needs the value of Ax from the agents. When the variables x are distributed into agents $x_1, \ldots, x_n$, and the equality constraints between the agents and the coordinator is expressed on a bipartite graph:

$$D_{ie}x_i - \bar{x}_e + z_{ie} = 0, (i, e) \in \mathcal{E}_2, \tag{57}$$

the communication of Ax and $B\bar{x} + z + y/\rho$ takes place in a distributed and parallel way, i.e., the ith agent obtains the information of $-\bar{x}_e + z_{ie} + y_{ie}/\rho$ for all e such that $(i, e) \in \mathcal{E}_2$ from the coordinator. The coordinator, based on inter-subsystem edges e in the digraph, obtains the information of $D_{ie}x_i$ for all related agents i. When the objective function and $\mathcal{X}$ are separable $f(x) = \sum_{i=1}^n f_i(x_i)$, $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$, based on such distributed and parallel communication, the optimization problem

$$\begin{aligned} \min \quad & f(x) - b\sum_{c=1}^{C_\phi} \ln\left(-\phi_c(x)\right) + \frac{\rho}{2}\left\|Ax + B\bar{x} + z + \frac{y}{\rho}\right\|^2 \\ \text{s.t.} \quad & \psi(x) = 0 \end{aligned} \tag{58}$$

in an x-update step can be solved in a distributed and parallel manner:

$$\left.\begin{array}{ll} \min_{x_i} & f_i(x_i) - b\sum_{c=1}^{C_{\phi,i}} \ln\left(-\phi_{c,i}(x_i)\right) + \frac{\rho}{2}\sum_{\{e|(i,e)\in\mathcal{E}_2\}} \left\| D_{ie}x_i - \bar{x}_e + z_{ie} + \frac{y_{ie}}{\rho}\right\|^2 \\ \text{s.t.} & \psi_i(x_i) = 0 \end{array}\right\} \text{// for } i. \tag{59}$$

Similarly, the $\bar{x}$-update with the G-mapping is in parallel for its components e, if $\bar{\mathcal{X}}$ is separable, i.e., if $\bar{\mathcal{X}}$ is a closed hypercube (whether bounded or unbounded), and if g is also separable. That is, $\bar{x}$-update can be expressed as

$$\left.\begin{array}{ll} \min_{\bar{x}_i} & g_i(\bar{x}_i) + \frac{\rho}{2}\sum_{\{i|(i,e)\in\mathcal{E}_2\}} \left\| D_{ie}x_i - \bar{x}_e + z_{ie} + \frac{y_{ie}}{\rho}\right\|^2 \\ \text{s.t.} & \bar{x}_i \in \bar{\mathcal{X}}_i \end{array}\right\} \text{// for } e. \tag{60}$$

The z and y updates are in parallel for the edges (i, e) on the bipartite graph.

In Algorithm 3, the procedures are written such that in each iteration, the update steps are carried out in sequence. This requires a synchronization of all the agents i and the coordinating elements e and (i, e). For example, for the x-update, every distributed agent needs to create a "finish" signal after solving x_i in (59) and send it to the coordinator. Only after the coordinator receives the "finish" signals from all the distributed agents can the $\bar{x}$-update be carried out. Due to the possible computational imbalance among the agents and the coordinator, such synchronization implies that faster updates must idle for some time to wait for slower ones. In fact, the convergence properties of the ELLADA algorithm do not rely on the synchronization. Even when the inner iterations are asynchronous, the update steps still contribute to the convergence of the barrier augmented Lagrangian and hence result in convergence to KKT conditions. The only exception is that under Anderson acceleration, the steps for generating the candidate of accelerated updates are allocated to another coordinator and another set of distributed agents, and they should communicate to make the decision on executing the accelerations.

6 Application to a quadruple tank process

The quadruple tank process is a simple benchmark process for distributed model predictive control (Johansson 2000) with 4 states (water heights in the 4 tanks) and 2 inputs (flow rates from the reservoir). The dynamic model is written as follows:

$$\begin{aligned} \dot{h}_1 &= -\frac{a_1}{A_1}\sqrt{h_1} + \frac{a_3}{A_1}\sqrt{h_3} + \frac{\gamma_1 k_1}{A_1}v_1, \dot{h}_2 &&= -\frac{a_2}{A_2}\sqrt{h_2} + \frac{a_4}{A_2}\sqrt{h_4} + \frac{\gamma_2 k_2}{A_2}v_2 \\ \dot{h}_3 &= -\frac{a_3}{A_3}\sqrt{h_3} + \frac{(1-\gamma_2)k_2}{A_3}v_2, && \dot{h}_4 = -\frac{a_4}{A_4}\sqrt{h_4} + \frac{(1-\gamma_1)k_1}{A_4}v_1. \end{aligned} \tag{61}$$

Other parameter values and the nominal steady state are given in Table 1.

The process is considered to have 2 subsystems, one containing tanks 1 and 4 and the other containing tanks 2 and 3. Each subsystem has 2 states, 1 input and 1

Table 1 Parameters and nominal steady state

Parameter	Value	Parameter	Value
A_1, A_3	28	a_1, a_3	3.145
A_2, A_4	32	a_2, a_4	2.525
γ_1	0.43	k_1	3.14
γ_2	0.34	k_2	3.29
Input	Value	Input	Value
v_1	3.15	v_2	3.15
State	Value	State	Value
h_1	12.44	h_2	13.17
h_3	4.73	h_4	4.99

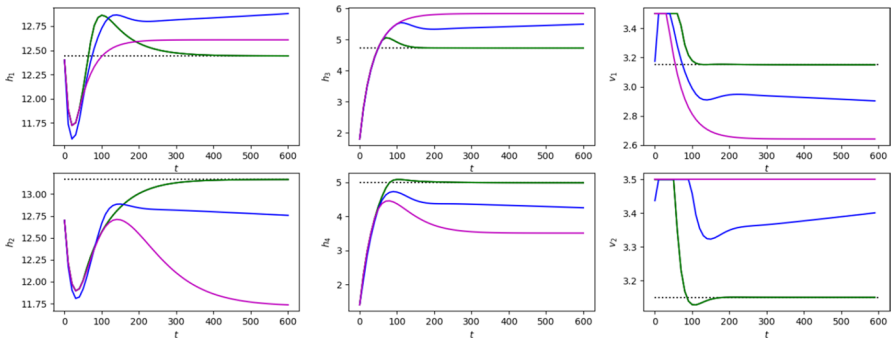

Fig. 6 Closed-loop trajectories under traditional MPC controllers for the quadruple tank process

upstream state. We first design a centralized MPC with quadratic objective function for each tank, and bounds on the inputs $2.5 \leq v_1, v_2 \leq 3.5$. We first decide through the simulation of centralized MPC that a receding horizon of $T = 400$ with sampling time $\delta t = 10$ is appropriate. The computations are performed using the Python module `pyomo.dae` with an IPOPT solver (Nicholson et al. 2018).

The closed-loop trajectories under the traditional MPC controllers, including a centralized MPC (black), a semi-centralized MPC where the inputs are iteratively updated based on predictions over the entire process (green), a decentralized MPC (blue), and a distributed MPC with only state feedforwarding among the agents (purple), are shown in Fig. 6. It was observed that a semi-centralized MPC based on system-wide prediction maintains the control performance, yielding trajectories overlapping with those of the centralized MPC. However, the state-feedforward distributed MPC without sufficient coordination accounting for the state interactions results in unsatisfactory control performance, whose ultimate deviation from the steady state is even larger than the decentralized MPC without any communication between the controllers.

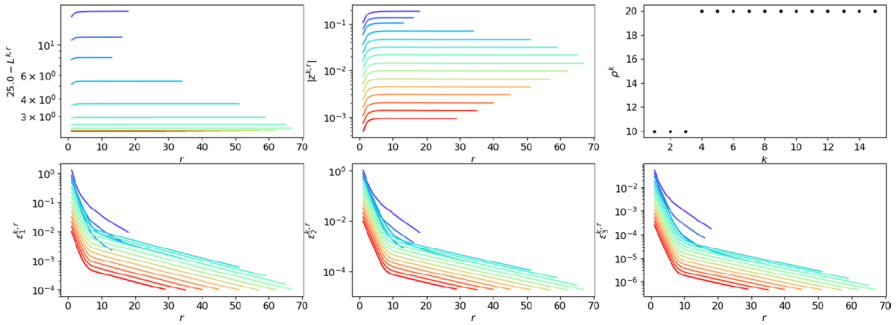

Fig. 7 Solution results of the ELL algorithm for the quadruple tank process

Next we use the proposed ELLADA algorithm for distributed nonlinear MPC of the process. We first examine the basic ELL algorithm (Algorithm 1) by solving the corresponding distributed MPC problem at a state with $h_1 = 12.6$, $h_2 = 12.4$, $h_3 = 5.0$, $h_4 = 4.5$, where we set $\omega = 0.75$, $\gamma = 2$, $\overline{\lambda} = -\underline{\lambda} = 10$ (in an element-wise sense) and $\epsilon_1^k = \epsilon_2^k = 10^{-2}/2^{k-1}$, $\epsilon_3^k = 10^{-1}/2^{k-1}$, $\epsilon_1 = \epsilon_2 = 10^{-4}$, $\epsilon_3 = 10^{-3}$ as a baseline that seeks to terminate only when a highly precise solution is reached. The solution results in terms of the variation of the augmented Lagrangian $L^{k,r}$, the violations to the KKT conditions $\epsilon_{1,2,3}^{k,r}$, and penalty parameters ρ^k throughout the inner and outer iterations are presented in Fig. 7, where the rainbow colormap from blue to red colors stand for increasing outer iteration number. In accordance to the conclusion of Lemma 1, the augmented Lagrangian is monotonically decreasing in each outer iterations and remains upper bounded, which guarantees the convergence of the algorithm. Using the ELL algorithm for the afore-mentioned closed-loop MPC simulation, the resulting trajectories are found identical to those of the centralized control, which corroborates the theoretical property of the algorithm of converging to the set of stationary solutions.

With the preserved control performance of the ELL algorithm, we seek to improve its computational efficiency with the ELLA and ELLADA algorithms (Algorithms 2 and 3). In ELLA, the tolerances for approximate NLP solution are tuned to $\epsilon_1 = \epsilon_2 = \epsilon_4 = 10^3\epsilon_3 = 1$, $\epsilon_1^k = \epsilon_2^k = 10^3\epsilon_3^k = \epsilon_4^k = 100/2^{k-1}$, $\epsilon_4^{k,r} = 10^3\epsilon_5^{k,r} = \max\left(\epsilon_4^k, 40\left(\epsilon_1^{k,r}\right)^2\right)$. The barrier constants are updated throughout outer iterations according to $\|z\|$ according to $b^{k+1} = \min\left(10^{-1}, \max\left(10^{-4}, 25\left(\epsilon_3^k\right)^2\right)\right)$. Compared to ELL, the accumulated number of iterations and computational time of ELLA are reduced by over an order of magnitude. To seek for better computational performance, we apply the ELLADA algorithm, where we set $M = 10$, $\sigma = 1$, $\eta_L = \eta_{\tilde{w}} = 0.01$, $\eta_\theta = 0.5$, $\eta_w = 0.05$. This further reduces the number of iterations and computational time. These results are shown in Fig. 8.

Compared to the basic ELL algorithm, ELLADA achieves acceleration by approximately 18 times in terms of iterations and 19 times in computational time for the entire simulation time span. These improvements are more significant when the

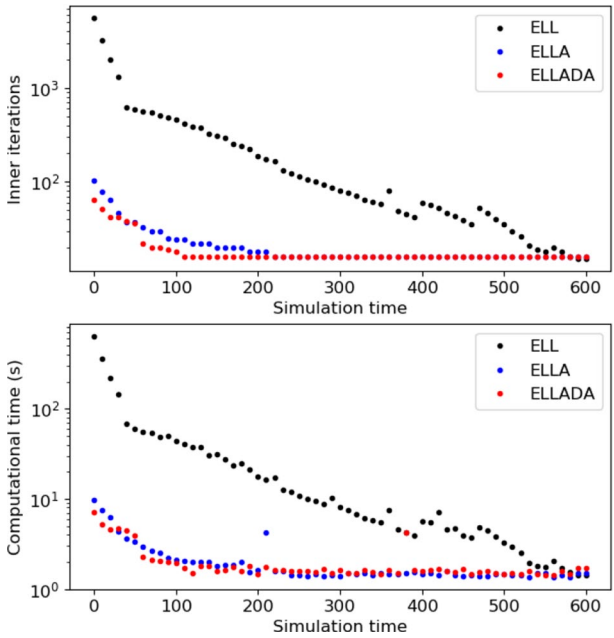

Fig. 8 Iteration and computational time under ELL, ELLA and ELLADA algorithms for the quadruple tank process

states are far from the target steady state (43 and 45 times, respectively, for the first 1/6 of the simulation). We note that the improvement from ELLA to ELLADA by using the Anderson scheme is not an order-of-magnitude one mainly because each outer iteration needs only a few number of inner iterations, leaving little space for further acceleration (e.g., for the first sampling time, 12 outer iterations including only 102 inner iterations are needed in ELLA, and in ELLADA, 61 inner iterations are needed). Under the accelerations, ELLADA returns the identical solution to the centralized optimization, thus preserving the control performance of the centralized MPC.

7 Conclusions and discussions

We have proposed a new algorithm for distributed optimization allowing nonconvex constraints, which simultaneously guarantees convergence under mild assumptions and achieves fast computation. Specifically, convergence is established by adopting a two-layer architecture. In the outer layer, the slack variables are tightened using the method of multipliers, and the inequalities are handled using a barrier technique. In the inner layer, ADMM iterations are performed in a distributed and coordinated manner. Approximate NLP solution and Anderson acceleration techniques are integrated into inner iterations for computational acceleration. A large-scale optimization problem from a NLP benchmark library is used to show the numerical

advantages of using the proposed ELLADA algorithm compared to the basic convergent algorithms ELL (Algorithm 1) and ELLA (Algorithm 2).

Distributed nonlinear MPC, as an challenging problem in process control, is an important application of such an algorithm. The advantages of applying ELLADA to distributed nonlinear MPC include:

- Arbitrary input and state couplings among subsystems are allowed. No specific pattern is required a priori.
- The convergence property of the algorithm towards a stationary point is theoretically guaranteed, and its performance can be monitored throughout iterations.
- Equality-constrained NLP solvers can be used only as a subroutine. No internal modification of solvers is needed, and the choice of any appropriate solver is flexible.
- Asynchronous updates are allowed without affecting the convergence properties.
- Although motivated with a nominal optimal control problem, the algorithm could be suitable for more intricate MPC formulations such as stochastic/robust MPC or sensitivity-based advance-step MPC.

The application of the ELLADA algorithm on the distributed nonlinear MPC of a quadruple tank process has already shown its improved computational performance compared to ELL and ELLA, and improved control performance compared to the decentralized MPC and distributed MPC without accounting for state interactions. Of course, due to the small size of the specific benchmark process, the control can be realized easily with a centralized MPC. A truly large-scale control problem is more suitable to demonstrate the effectiveness of our algorithm, and this shall be presented in an upcoming separate paper.

Acknowledgements This work was supported by National Science Foundation (NSF-CBET). The authors would also like to thank Prof. Qi Zhang for his constructive opinions.

Appendix 1: Proof of Lemma 1

We first prove that

$$\begin{aligned} L\left(x^{k,r+1},\bar{x}^{k,r+1},z^{k,r+1},y^{k,r+1}\right) &\le L\left(x^{k,r},\bar{x}^{k,r},z^{k,r},y^{k,r}\right) \\ &- \beta^k\left\|B\bar{x}^{k,r+1}-B\bar{x}^{k,r}\right\|^2-\frac{\beta^k}{2}\left\|z^{k,r+1}-z^{k,r}\right\|^2 \end{aligned} \tag{62}$$

for $r=0,1,2,\ldots$. First, since $x^{k,r+1}$ is chosen as the minimizer of the augmented Lagrangian with respect to x (Line 9, Algorithm 1), the update of x leads to a decrease in L:

$$L\left(x^{k,r+1},\bar{x}^{k,r},z^{k,r},y^{k,r}\right)\le L\left(x^{k,r},\bar{x}^{k,r},z^{k,r},y^{k,r}\right). \tag{63}$$

Then consider the decrease resulted from $\bar{x}$-update:

$$
\begin{aligned}
&L\left(x^{k,r+1},\bar{x}^{k,r+1},z^{k,r},y^{k,r}\right)-L\left(x^{k,r+1},\bar{x}^{k,r},z^{k,r},y^{k,r}\right)\\
&\quad=g\left(\bar{x}^{k,r+1}\right)-g\left(\bar{x}^{k,r}\right)+y^{k,r\top}\left(B\bar{x}^{k,r+1}-B\bar{x}^{k,r}\right)\\
&\qquad+\frac{\rho^k}{2}\left\|Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r}\right\|^2-\frac{\rho^k}{2}\left\|Ax^{k,r+1}+B\bar{x}^{k,r}+z^{k,r}\right\|^2\\
&\quad=g\left(\bar{x}^{k,r+1}\right)-g\left(\bar{x}^{k,r}\right)-\frac{\rho^k}{2}\left\|B\bar{x}^{k,r+1}-B\bar{x}^{k,r}\right\|^2\\
&\qquad-\rho^k\left(\bar{x}^{k,r}-\bar{x}^{k,r+1}\right)^\top B^\top\left(Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r}+\frac{y^{k,r}}{\rho^k}\right).
\end{aligned}
\tag{64}
$$

The minimization of $\bar{x}$ (Line 10, Algorithm 1) should satisfy the optimality condition

$$
0\in\rho^kB^\top\left(Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r}+\frac{y^{k,r}}{\rho^k}\right)+\partial g\left(\bar{x}^{k,r+1}\right)+\mathcal{N}_{\bar{\mathcal{X}}}\left(\bar{x}^{k,r+1}\right),
\tag{65}
$$

i.e., there exist vectors $v_1\in\partial g\left(\bar{x}^{k,r+1}\right)$ and $v_2\in\mathcal{N}_{\bar{\mathcal{X}}}\left(\bar{x}^{k,r+1}\right)$ with

$$
\rho^kB^\top\left(Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r}+\frac{y^{k,r}}{\rho^k}\right)=-v_1-v_2.
\tag{66}
$$

Since $v_1\in\partial g\left(\bar{x}^{k,r+1}\right)$ and g is convex, $v_1^\top\left(\bar{x}^{k,r}-\bar{x}^{k,r+1}\right)\le g\left(\bar{x}^{k,r}\right)-g\left(\bar{x}^{k,r+1}\right)$. And $v_2\in\mathcal{N}_{\bar{\mathcal{X}}}\left(\bar{x}^{k,r+1}\right)$ implies $v_2^\top\left(\bar{x}^{k,r}-\bar{x}^{k,r+1}\right)\le 0$. Hence

$$
\begin{aligned}
&\rho^k\left(\bar{x}^{k,r}-\bar{x}^{k,r+1}\right)^\top B^\top\left(Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r}+\frac{y^{k,r}}{\rho^k}\right)\\
&\quad=-v_1^\top\left(\bar{x}^{k,r}-\bar{x}^{k,r+1}\right)-v_2^\top\left(\bar{x}^{k,r}-\bar{x}^{k,r+1}\right)\\
&\quad\ge-\left(g\left(\bar{x}^{k,r}\right)-g\left(\bar{x}^{k,r+1}\right)\right).
\end{aligned}
\tag{67}
$$

Substituting the above inequality in (64), we obtain

$$
\begin{aligned}
&L\left(x^{k,r+1},\bar{x}^{k,r+1},z^{k,r},y^{k,r}\right)\le L\left(x^{k,r+1},\bar{x}^{k,r},z^{k,r},y^{k,r}\right)\\
&\quad-\frac{\rho^k}{2}\left\|B\bar{x}^{k,r+1}-B\bar{x}^{k,r}\right\|^2.
\end{aligned}
\tag{68}
$$

Third, we consider the decrease resulted from z- and y-updates:

$$\begin{aligned}
&L\left(x^{k,r+1},\bar{x}^{k,r+1},z^{k,r+1},y^{k,r+1}\right)-L\left(x^{k,r+1},\bar{x}^{k,r+1},z^{k,r},y^{k,r}\right)\\
&\quad=\lambda^{k\top}\left(z^{k,r+1}-z^{k,r}\right)+\frac{\beta^k}{2}\left(\left\|z^{k,r+1}\right\|^2-\left\|z^{k,r}\right\|^2\right)\\
&\qquad+y^{k,r+1\top}\left(Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r+1}\right)\\
&\qquad-y^{k,r\top}\left(Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r}\right)\\
&\qquad+\frac{\rho^k}{2}\left\|Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r+1}\right\|^2\\
&\qquad-\frac{\rho^k}{2}\left\|Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r}\right\|^2.
\end{aligned}\tag{69}$$

Since $v(z;\lambda,\beta)=\lambda^\top z+\frac{\beta}{2}\|z\|^2$ is a convex function, whose gradient is $\nabla v(z;\lambda,\beta)=\lambda+\beta z$,

$$v\left(z^{k,r+1};\lambda^k,\beta^k\right)-v\left(z^{k,r};\lambda^k,\beta^k\right)\le\left(\lambda^k+\beta^k z^{k,r+1}\right)^\top\left(z^{k,r+1}-z^{k,r}\right),\tag{70}$$

From Line 11 of Algorithm 1 it can be obtained

$$\lambda^k+\beta z^{k,r+1}=-y^{k,r+1}.\tag{71}$$

Substituting into (69), we obtain

$$\begin{aligned}
&L\left(x^{k,r+1},\bar{x}^{k,r+1},z^{k,r+1},y^{k,r+1}\right)-L\left(x^{k,r+1},\bar{x}^{k,r+1},z^{k,r},y^{k,r}\right)\\
&\le\left(y^{k,r+1}-y^{k,r}\right)^\top\left(Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r}\right)\\
&\quad+\frac{\rho^k}{2}\left\|Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r+1}\right\|^2-\frac{\rho^k}{2}\left\|Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r}\right\|^2\\
&=\frac{\rho^k}{2}\left(Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r+1}\right)^\top\left(Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r}\right)\\
&\quad+\frac{\rho^k}{2}\left\|Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r+1}\right\|^2-\frac{\rho^k}{2}\left\|Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r}\right\|^2\\
&=-\frac{\rho^k}{2}\left\|z^{k,r+1}-z^{k,r}\right\|^2+\rho^k\left\|Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r+1}\right\|^2
\end{aligned}\tag{72}$$

From (71),

$$Ax^{k,r+1}+B\bar{x}^{k,r+1}+z^{k,r+1}=\frac{1}{\rho_k}\left(y^{k,r+1}-y^{k,r}\right)=-\frac{\beta^k}{\rho^k}\left(z^{k,r+1}-z^{k,r}\right).\tag{73}$$

Then (72) becomes

$$\begin{aligned}
&L\left(x^{k,r+1},\bar{x}^{k,r+1},z^{k,r+1},y^{k,r+1}\right)-L\left(x^{k,r+1},\bar{x}^{k,r+1},z^{k,r},y^{k,r}\right)\\
&\le-\left(\frac{\rho^k}{2}-\frac{\left(\beta^k\right)^2}{\rho^k}\right)\left\|z^{k,r+1}-z^{k,r}\right\|^2=-\frac{\beta^k}{2}\left\|z^{k,r+1}-z^{k,r}\right\|^2.
\end{aligned}\tag{74}$$

Summing up the inequalities (63), (68) and (74), we have proved the inequality (62).

Next, we show that the augmented Lagrangian is lower bounded, and hence is convergent towards some $\underline{L}^k \in \mathbb{R}$. We note that $\upsilon(z;\lambda,\beta)$ is a convex function of modulus β, it can be easily verified that

$$\begin{aligned}\upsilon\left(z^{k,r};\lambda^k,\beta^k\right)+\left(\lambda^k+\beta^k z^{k,r}\right)^\top\left(z'-z^{k,r}\right)\\+\frac{\rho^k}{2}\left\|z'-z^{k,r}\right\|^2\geq\upsilon\left(z';\lambda^k,\beta^k\right)\end{aligned}\tag{75}$$

for any z', i.e.,

$$\upsilon\left(z^{k,r};\lambda^k,\beta^k\right)+y^{k,r\top}\left(z^{k,r}-z'\right)\geq\upsilon\left(z';\lambda^k,\beta^k\right)-\frac{\rho^k}{2}\left\|z'-z^{k,r}\right\|^2.\tag{76}$$

Let $z'=-\left(Ax^{k,r}+B\bar{x}^{k,r}\right)$ and remove the last term on the right-hand side. Then

$$\begin{aligned}\upsilon\left(z^{k,r};\lambda^k,\beta^k\right)+y^{k,r\top}\left(Ax^{k,r}+B\bar{x}^{k,r}+z^{k,r}\right)\\\geq\upsilon\left(-\left(Ax^{k,r}+B\bar{x}^{k,r}\right);\lambda^k,\beta^k\right).\end{aligned}\tag{77}$$

Hence

$$\begin{aligned}&L\left(x^{k,r},\bar{x}^{k,r+1},z^{k,r},y^{k,r}\right)\\&=f\left(x^{k,r}\right)+g\left(\bar{x}^{k,r}\right)+\upsilon\left(z^{k,r};\lambda^k,\beta^k\right)\\&\quad+y^{k,r\top}\left(Ax^{k,r}+B\bar{x}^{k,r}+z^{k,r}\right)+\frac{\rho^k}{2}\left\|Ax^{k,r}+B\bar{x}^{k,r}+z^{k,r}\right\|^2\\&\geq f\left(x^{k,r}\right)+g\left(\bar{x}^{k,r}\right)+\upsilon\left(-\left(Ax^{k,r}+B\bar{x}^{k,r}\right);\lambda^k,\beta^k\right).\end{aligned}\tag{78}$$

Since $\upsilon(z)=\lambda^\top z+\frac{\beta}{2}\|z\|^2\geq-\|\lambda\|^2/(2\beta)$, λ is bounded in $\left[\underline{\lambda},\overline{\lambda}\right]$, $\beta^k\geq\beta^1$, and f and g are bounded below, L has a lower bound.

Taking the limit $r\to\infty$ on the both sides of inequality (62), it becomes obvious that $B\bar{x}^{k,r+1}-B\bar{x}^{k,r}$ and $z^{k,r+1}-z^{k,r}$ converge to 0. Due to (73), we have $Ax^{k,r}+B\bar{x}^{k,r}+z^{k,r}\to 0$. Hence there must exist a r such that (22) is met. At this time, the optimality conditions for $x^{k,r+1}$ is written as

$$0\in\partial f\left(x^{k,r+1}\right)+\mathcal{N}_{\mathcal{X}}\left(x^{k,r+1}\right)+A^\top y^{k,r}+\rho^k A^\top\left(Ax^{k,r+1}+B\bar{x}^{k,r}+z^{k,r}\right).\tag{79}$$

According to the update rule of $y^{k,r}$, the above expression is equivalent to

$$\begin{aligned}0\in&\partial f\left(x^{k,r+1}\right)+\mathcal{N}_{\mathcal{X}}\left(x^{k,r+1}\right)+A^\top y^{k,r+1}-\rho^k A^\top\\&\left(B\bar{x}^{k,r+1}+z^{k,r+1}-B\bar{x}^{k,r}-z^{k,r}\right),\end{aligned}\tag{80}$$

i.e.,

$$\begin{aligned}\rho^k A^\top\left(B\bar{x}^{k,r+1}+z^{k,r+1}-B\bar{x}^{k,r}-z^{k,r}\right)\in\partial f\left(x^{k,r+1}\right)\\+\mathcal{N}_{\mathcal{X}}\left(x^{k,r+1}\right)+A^\top y^{k,r+1}.\end{aligned}\tag{81}$$

According to the first inequality of (22), the norm of the left hand side above is not larger than ϵ_1^k, which directly implies the first condition in (23). In a similar manner, the second condition in (23) can be established. The third one follows from (71) and the fourth condition is obvious.

Appendix 2: Proof of Lemma 2

We first consider the situation when β^k is unbounded. From (78), we have

$$\overline{L} \geq f\left(x^{k+1}\right) + g\left(x^{k+1}\right) - \lambda^{k\top}\left(Ax^{k+1} + B\bar{x}^{k+1}\right) + \frac{\beta^k}{2}\left\|Ax^{k+1} + B\bar{x}^{k+1}\right\|^2. \tag{82}$$

Since f and g are both lower bounded, as $\beta^k \to \infty$, we have $Ax^{k+1} + B\bar{x}^{k+1} \to 0$. Combined with the first two conditions of (23) in the limit of $\epsilon_1^k, \epsilon_2^k, \epsilon_3^k \downarrow 0$, we have reached (25).

Then we suppose that β^k is bounded, i.e., the amplification step $\beta^{k+1} = \gamma\beta^k$ is executed for only a finite number of outer iterations. According to Lines 17–21 of Algorithm 1, expect for some finite choices of k, $\left\|z^{k+1}\right\| \leq \omega\left\|z^k\right\|$ always hold. Therefore $z^{k+1} \to 0$. Apparently, (25) follows from the limit of (23).

Appendix 3: Proof of Lemma 3

From Lemma 1 one knows that within R inner iterations

$$\frac{\overline{L} - \underline{L}^k}{\beta^k} \geq \sum_{r=1}^{R}\left(\left\|B\bar{x}^{k,r+1} - B\bar{x}^{k,r}\right\|^2 + \frac{1}{2}\left\|\tilde{z}^{k,r+1} - z^{k,r}\right\|^2\right). \tag{83}$$

Then

$$\left\|B\bar{x}^{k,R+1} - B\bar{x}^{k,R}\right\|, \left\|z^{k,R+1} - z^{k,R}\right\| \sim \mathcal{O}\left(1/\sqrt{\beta^k R}\right). \tag{84}$$

For the kth outer iteration, its inner iterations are terminated when (22) is met, which is translated into the following relations:

$$\begin{aligned} &\mathcal{O}\left(\rho^k/\sqrt{\beta^k R^k}\right) \leq \epsilon_1^k, \epsilon_2^k \sim \mathcal{O}\left(\vartheta^k\right), \\ &\mathcal{O}\left(1/\sqrt{\beta^k R^k}\right) \leq \epsilon_3^k \sim \mathcal{O}\left(\vartheta^k/\beta^k\right). \end{aligned} \tag{85}$$

where the last relation uses (73) with $\rho^k = 2\beta^k$. Therefore

$$R^k \sim \mathcal{O}\left(\beta^k/\vartheta^{2k}\right). \tag{86}$$

At the end of the kth iteration, suppose that Lines 19–20 and Lines 17–18 of Algorithm 1 have been executed for k_1 and k_2 times,

respectively ($k_1 + k_2 = k$). Then the obtained z^{k+1} satisfies $\|z^{k+1}\| \sim \mathcal{O}(\omega^{k_1})$, and $\|Ax^{k+1} + B\bar{x}^{k+1} + z^{k+1}\| \leq \epsilon_3^k \sim \mathcal{O}(\vartheta^k/\beta^k)$, which imply

$$\left\|Ax^{k+1} + B\bar{x}^{k+1}\right\| \leq \mathcal{O}(\vartheta^k/\beta^k) + \mathcal{O}(\omega^{k_1}). \tag{87}$$

From (82),

$$\beta^k \left\|Ax^{k+1} + B\bar{x}^{k+1}\right\|^2 \sim \beta^k \left(\mathcal{O}(\vartheta^k/\beta^k) + \mathcal{O}(\omega^{k_1})\right)^2 \sim \mathcal{O}(1). \tag{88}$$

Substituting (88) into (86), we obtain

$$R^k \sim \mathcal{O}\left(\frac{1}{\vartheta^{2k}} \frac{1}{\left(\mathcal{O}(\vartheta^k/\beta^k) + \mathcal{O}(\omega^{k_1})\right)^2}\right). \tag{89}$$

When $\vartheta \leq \omega$, $\vartheta^k \leq \omega^k \leq \omega^{k_1}\gamma^{k_2}$, and hence $\gamma^{k_2}\vartheta^k \leq \omega^{k_1}$, i.e., ω^{k_1} dominates over ϑ^k/β^k, leading to

$$R^k \sim \mathcal{O}(1/\vartheta^{2k}\omega^{2k_1}) \sim \mathcal{O}(1/\vartheta^{2k}\omega^{2k}). \tag{90}$$

For K outer iterations, the total number of inner iterations is

$$R = \sum_{k=1}^{K} R^k \sim \mathcal{O}\left(\sum_{k=1}^{K} \frac{1}{\vartheta^{2k}\omega^{2k}}\right) \sim \mathcal{O}\left(\frac{1}{\vartheta^{2K}\omega^{2K}}\right). \tag{91}$$

The number of outer iterations needed to reach an ϵ-approximate stationary point is obviously $K \sim \mathcal{O}(\log_\vartheta \epsilon)$. Then

$$R \sim \mathcal{O}(\epsilon^{-2(1+\varsigma)}). \tag{92}$$

Appendix 4: Proof of Lemma 6

Through the inner iterations, only Anderson acceleration might lead to an increase in the barrier augmented Lagrangian. Combining Assumptions 3, 5, and the safeguarding criterion (41), we obtain

$$L_{b^k}\left(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r+1}, y^{k,r+1}\right) \leq \bar{L} + \tilde{L}_0\eta_L \sum_{r=0}^{\infty} \frac{1}{r^{1+\sigma}} < +\infty, \tag{93}$$

Together with Assumptions 1 and 2, L_{b^k} is also bounded below. Therefore L_{b^k} is bounded in a closed interval and must have converging subsequences. Therefore we can choose a subsequence converging to the lower limit $\underline{L}$. For any $\varepsilon > 0$ there exists an index R of inner iteration in this subsequence, such that $\tilde{L}_0\eta_L \sum_{r=R}^{\infty} r^{-(1+\sigma)} < \varepsilon/2$ and $L_{b^k}\left(x^{k,r+1}, \bar{x}^{k,r+1}, z^{k,r+1}, y^{k,r+1}\right) < \underline{L} + \varepsilon/2$ for any $r \geq R$ on this subsequence. It then follows that for any $r \geq R$, whether on the subsequence or not, it holds that

$$L_{b^k}\left(x^{k,r+1},\bar{x}^{k,r+1},z^{k,r+1},y^{k,r+1}\right)<\underline{L}+\varepsilon. \tag{94}$$

Hence the upper limit is not larger than $\underline{L}+\varepsilon$. Due to the arbitrariness of $\varepsilon>0$, the lower limit coincides with the upper limit, and hence the sequence of barrier augmented Lagrangian is convergent.

The convergence of the barrier augmented Lagrangian implies that as $r\to\infty$,

$$L_{b^k}\left(x^{k,r+1},\bar{x}^{k,r+1},z^{k,r+1},y^{k,r+1}\right)-L_{b^k}\left(x^{k,r},\bar{x}^{k,r},z^{k,r},y^{k,r}\right)\to 0. \tag{95}$$

Suppose that r is not an accelerated iteration, then since this quantity does not exceed $-\beta^k\left\|B\bar{x}^{k,r+1}-B\bar{x}^{k,r}\right\|^2-\left(\beta^k/2\right)\left\|z^{k,r+1}-z^{k,r}\right\|^2$, we must have $B\bar{x}^{k,r+1}-B\bar{x}^{k,r}\to 0$ and $z^{k,r+1}-z^{k,r}\to 0$. Otherwise if inner iteration r is accelerated, the convergence of $B\bar{x}^{k,r+1}-B\bar{x}^{k,r}$ and $z^{k,r+1}-z^{k,r}$ are automatically guaranteed by the second criterion (42) of accepting Anderson acceleration. The convergence properties of these two sequences naturally fall into the paradigm of Lemma 1 for establishing the convergence to approximate KKT conditions of the relaxed problem.

References

Anderson DG (1965) Iterative procedures for nonlinear integral equations. J ACM 12(4):547–560

Bertsekas DP (2016) Nonlinear programming, 3rd edn. Athena Scientific, Nashua

Biegler LT, Thierry DM (2018) Large-scale optimization formulations and strategies for nonlinear model predictive control. IFAC-PapersOnLine 51(20):1–15, the 6th IFAC Conference on Nonlinear Model Predictive Control (NMPC)

Boyd S, Parikh N, Chu E, Peleato B, Eckstein J (2011) Distributed optimization and statistical learning via the alternating direction method of multipliers. Found Trend Mach Learn 3(1):1–122

Chen X, Heidarinejad M, Liu J, Christofides PD (2012) Distributed economic MPC: application to a nonlinear chemical process network. J Process Control 22(4):689–699

Chen C, He B, Ye Y, Yuan X (2016) The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent. Math Prog 155(1–2):57–79

Christofides PD, Scattolini R, Muñoz de la Peña D, Liu J (2013) Distributed model predictive control: a tutorial review and future research directions. Comput Chem Eng 51:21–41

Daoutidis P, Tang W, Jogwar SS (2018) Decomposing complex plants for distributed control: perspectives from network theory. Comput Chem Eng 114:43–51

Daoutidis P, Tang W, Allman A (2019) Decomposition of control and optimization problems by network structure: concepts, methods and inspirations from biology. AIChE J 65(10):e16708

Dhingra NK, Khong SZ, Jovanović MR (2019) The proximal augmented Lagrangian method for nonsmooth composite optimization. IEEE Trans Autom Control 64(7):2861–2868

Dolan ED, Moré JJ (2002) Benchmarking optimization software with performance profiles. Math Prog 91(2):201–213

Eckstein J, Bertsekas DP (1992) On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. Math Prog 55(1–3):293–318

Eckstein J, Yao W (2017) Approximate ADMM algorithms derived from Lagrangian splitting. Comput Optim Appl 68(2):363–405

Eckstein J, Yao W (2018) Relative-error approximate versions of Douglas-Rachford splitting and special cases of the ADMM. Math Prog 170(2):417–444

Fang Hr, Saad Y (2009) Two classes of multisecant methods for nonlinear acceleration. Numer Linear Algebra Appl 16(3):197–221

Farokhi F, Shames I, Johansson KH (2014) Distributed MPC via dual decomposition and alternative direction method of multipliers. In: Distributed model predictive control made easy. Springer, Berlin, pp 115–131

Fu A, Zhang J, Boyd S (2019) Anderson accelerated Douglas–Rachford splitting. arXiv preprint arXiv:190811482

Gabay D, Mercier B (1976) A dual algorithm for the solution of nonlinear variational problems via finite element approximation. Comput Math Appl 2(1):17–40

Giselsson P, Doan MD, Keviczky T, De Schutter B, Rantzer A (2013) Accelerated gradient methods and dual decomposition in distributed model predictive control. Automatica 49(3):829–833

Glowinski R, Marroco A (1975) Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires. Rev Fr Autom Inform Rech Opér, Anal Numér 9(R2):41–76

Goldstein T, O'Donoghue B, Setzer S, Baraniuk R (2014) Fast alternating direction optimization methods. SIAM J Imaging Sci 7(3):1588–1623

Hajinezhad D, Hong M (2019) Perturbed proximal primal–dual algorithm for nonconvex nonsmooth optimization. Math Prog 176(1–2):207–245

He B, Yuan X (2012) On the $O(1/n)$ convergence rate of the Douglas–Rachford alternating direction method. SIAM J Numer Anal 50(2):700–709

Hong M, Luo ZQ (2017) On the linear convergence of the alternating direction method of multipliers. Math Prog 162(1–2):165–199

Hong M, Luo ZQ, Razaviyayn M (2016) Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. SIAM J Optim 26(1):337–364

Hours JH, Jones CN (2015) A parametric nonconvex decomposition algorithm for real-time and distributed NMPC. IEEE Trans Autom Control 61(2):287–302

Houska B, Frasch J, Diehl M (2016) An augmented Lagrangian based algorithm for distributed nonconvex optimization. SIAM J Optim 26(2):1101–1127

Jalving J, Cao Y, Zavala VM (2019) Graph-based modeling and simulation of complex systems. Comput Chem Eng 125:134–154

Jiang B, Lin T, Ma S, Zhang S (2019) Structured nonconvex and nonsmooth optimization: algorithms and iteration complexity analysis. Comput Optim Appl 72(1):115–157

Johansson KH (2000) The quadruple-tank process: a multivariable laboratory process with an adjustable zero. IEEE Trans Control Syst Technol 8(3):456–465

Li G, Pong TK (2015) Global convergence of splitting methods for nonconvex composite optimization. SIAM J Optim 25(4):2434–2460

Liu J, Chen X, Muñoz de la Peña D, Christofides PD (2010) Sequential and iterative architectures for distributed model predictive control of nonlinear process systems. AIChE J 56(8):2137–2149

Mota JF, Xavier JM, Aguiar PM, Püschel M (2014) Distributed optimization with local domains: applications in MPC and network flows. IEEE Trans Autom Control 60(7):2004–2009

Nesterov YuE (1983) A method of solving a convex programming problem with convergence rate $O(\frac{1}{k^2})$. Dokl Akad Nauk SSSR 269(3):543–547

Nicholson B, Siirola JD, Watson JP, Zavala VM, Biegler LT (2018) `pyomo.dae`: a modeling and automatic discretization framework for optimization with differential and algebraic equations. Math Prog Comput 10(2):187–223

Nishihara R, Lessard L, Recht B, Packard A, Jordan M (2015) A general analysis of the convergence of ADMM. Proc Mach Learn Res 37:343–352

Ouyang Y, Chen Y, Lan G, Pasiliao E Jr (2015) An accelerated linearized alternating direction method of multipliers. SIAM J Imaging Sci 8(1):644–681

Patterson MA, Rao AV (2014) GPOPS-II: a MATLAB software for solving multiple-phase optimal control problems using h_p-adaptive Gaussian quadrature collocation methods and sparse nonlinear programming. ACM Trans Math Softw (TOMS) 41(1):1–37

Pulay P (1980) Convergence acceleration of iterative sequences. The case of SCF iteration. Chem Phys Lett 73(2):393–398

Rawlings JB, Mayne DQ, Diehl MM (2017) Model predictive control: theory, computation, and design, 2nd edn. Nob Hill Publishing, Madison

Rockafellar RT, Wets RJB (1998) Variational analysis. Springer, Berlin

Scattolini R (2009) Architectures for distributed and hierarchical model predictive control—a review. J Process Control 19(5):723–731

Scutari G, Facchinei F, Lampariello L (2016) Parallel and distributed methods for constrained nonconvex optimization—part I: theory. IEEE Trans Signal Process 65(8):1929–1944

Stewart BT, Venkat AN, Rawlings JB, Wright SJ, Pannocchia G (2010) Cooperative distributed model predictive control. Syst Control Lett 59(8):460–469

Sun K, Sun XA (2019) A two-level distributed algorithm for general constrained non-convex optimization with global convergence. arXiv preprint arXiv:190207654

Tang W, Allman A, Pourkargar DB, Daoutidis P (2018) Optimal decomposition for distributed optimization in nonlinear model predictive control through community detection. Comput Chem Eng 111:43–54

Themelis A, Patrinos P (2020) Douglas-Rachford splitting and ADMM for nonconvex optimization: tight convergence results. SIAM J Optim 30(1):149–181

Toth A, Kelley C (2015) Convergence analysis for Anderson acceleration. SIAM J Numer Anal 53(2):805–819

Wächter A, Biegler LT (2005) Line search filter methods for nonlinear programming: motivation and global convergence. SIAM J Optim 16(1):1–31

Wächter A, Biegler LT (2006) On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math Prog 106(1):25–57

Wang Y, Boyd S (2009) Fast model predictive control using online optimization. IEEE Trans Control Syst Technol 18(2):267–278

Wang Z, Ong CJ (2017) Distributed model predictive control of linear discrete-time systems with local and global constraints. Automatica 81:184–195

Wang Y, Yin W, Zeng J (2019) Global convergence of ADMM in nonconvex nonsmooth optimization. J Sci Comput 78(1):29–63

Xie J, Liao A, Yang X (2017) An inexact alternating direction method of multipliers with relative error criteria. Optim Lett 11(3):583–596

Yang Y, Hu G, Spanos CJ (2020) A proximal linearization-based fecentralized method for nonconvex problems with nonlinear constraints. arXiv preprint arXiv:200100767

Zhang RY, White JK (2018) GMRES-accelerated ADMM for quadratic objectives. SIAM J Optim 28(4):3025–3056

Zhang J, O'Donoghue B, Boyd S (2018) Globally convergent type-I Anderson acceleration for nonsmooth fixed-point iterations. arXiv preprint arXiv:180803971

Zhang J, Peng Y, Ouyang W, Deng B (2019) Accelerating ADMM for efficient simulation and optimization. ACM Trans Graph 38(6):163