Asynchronous ADMM via a Data Exchange Server

ZhouDan Pan and Mark Cannon

***Abstract*— With advances in inter-processing-unit communication technology, distributed algorithms are becoming increasingly advantageous. This paper focuses on solving convex distributed optimisation problems with local consensus coupling constraints via the alternating direction method of multipliers (ADMM), by means of an asynchronous methodology allowing for communication delays. We use a bipartite undirected graph to denote the update structure of the processing agents that cooperatively perform the distributed algorithm without a centralised aggregator. We introduce a data server to exchange the asynchronous consensus data among the processing agents. Under certain technical assumptions that involve bounded delays, bounded step sizes, and strong convexities in parts of the local objectives, the running average of the local iterates generated by the proposed asynchronous algorithm converge to an optimal solution.**

I. Introduction

Recent advances in communication technologies and embedded systems have motivated the development of algorithms to coordinate intelligent agents in a distributed fashion. In contrast to centralised decision-making mechanisms, distributed optimisation algorithms [1]–[3] feature participating agents iteratively solving local optimisation problems and sharing information with neighbouring agents, as specified by an update and communication protocol. Such protocols can be designed to facilitate the solution of large-scale problems with privacy intrinsically protected. The optimal solution of the global problem is asymptotically obtained from the local iterates. However challenges may arise as a result of limited local computational power and communication reliability.

In general, distributed optimisation algorithms are designed to iteratively approach the optimal solution either by means of primal or primal-dual iterations. Motivated by [4], a class of distributed subgradient methods collaboratively estimates the common primal consensus via weighted averaging of local objective subgradient updates over a possibly time-varying network, with the capability to have global or local constraints [5], delays [6] and asymmetrical communication [7]. As the problem scales up, to keep a local copy of the entire consensus vector for every agent becomes prohibitive, hence dual decomposition algorithms [8] which allow agents to share only local variables become more advantageous. The alternating direction method of multipliers (ADMM) [1], [9] empowers the dual decomposition with an augmented Lagrangian to enhance the class of problems that can be tackled. For a convergence rate analysis we refer to [10]–[13].

The authors are with Department of Engineering Science, University of Oxford, Parks Road, Oxford OX1 3PJ, UK. zhoudan.pan, mark.cannon @eng.ox.ac.uk.

As the number of agents increases, with delays becoming more prevalent due to distantness, packet congestion and limited availability or capability of processors, synchronous algorithms may be vitiated by a single straggler. In [14], the effectiveness of distributed machine learning over a stale synchronous server was discussed. This motivates us to explore distributed optimisation algorithms with delays. Instead of waiting until all agents are synchronised at each iterative step, each agent uses the most recent information at hand to compute the next update, hence achieving better efficiency with respect to the diminishing overall waiting time [15], [16]. However such optimisation algorithms have a natural limitation since the outdated data employed at each update step may cause an undesirable accumulation of error in solution estimates. The work of [15] shows that in general fixed-point algorithms such a trade-off can be favourable.

A. Related Work

The works [17], [18] investigate asynchronous optimisation algorithms applied to a collection of gradient-like methods. In [19] the authors focus on the delayed sub-gradient method performed by a centralised coordinator, and in the later work [20] it is extended with an averaging consensus algorithm. The work of [21] extends these developments to stochastic convex optimisation problems. The dual gradient method for asynchronous distributed optimisation is explored in [22]. A framework for the convergence analyses of asynchronous fixed-point distributed optimisation algorithms is provided by [15], [23]. Addressing the need for parallel computing algorithms in the field of machine learning, [24], [25] study delay-tolerant gradient algorithms for distributed learning.

In this paper we focus on asynchronous distributed optimisation via ADMM [16], [26]–[37]. The work of [26] studies ADMM with asynchronous updates and relates the almost sure convergence property to the case of synchronous ADMM. Randomised ADMM is introduced in [27] with randomised Gauss-Seidel iterations and convergence analysis via non-expansiveness. In [28] the authors propose an asynchronous ADMM algorithm with a centralised aggregator, and also provide an intuitive explanation for the convergence of the respective expected values provided that the agents have equal probability delivering the updates to the aggregator. Based on similar theoretical analyses, the works [29], [31], [33] propose hierarchical communication strategies for asynchronous

ADMM. The works [16], [34]–[36] explore asynchronous ADMM with a centralised aggregator, and propose three algorithms whose convergence analyses are based on worst case bounded delay scenarios, to which our proposed algorithm is closely related. In [30] the authors propose a proximal and majorized approximation variant of ADMM, while the work [37] presents an incremental delayed-gradient variant, to enable the aggregator to cope with asynchrony and non-convexity.

Recent studies [38]–[42] investigate the application of distributed optimization algorithms through ADMM, thus eliminating the need for a centralized aggregator. In [38], an averaging algorithm is used to achieve a consensus of the global primal residual and dual variable, thereby replacing the centralized aggregator. [41] presents a different approach, introducing a pairwise structure that is employed to compute the bridging copies of local variables for relaxed ADMM. This method address problems that have a common global decision variable while also permitting asynchronous updates with probabilistic convergence. Further, [39] integrates an inner loop of a directed averaging algorithm. This strategy allows the distributed computation of the ϵ-consensus of the global decision variable. Similarly, [40] introduces an inner loop but for computing the finite-time exact ratio consensus. Finally, [42] tackles a bipartite optimal power flow problem with asynchrony via ADMM, utilizing learning algorithms to create replacements for missing updates.

B. Contribution

In this paper we propose a decentralised asynchronous communication and update protocol that uses ADMM to solve a convex optimisation problem comprising two groups of local cost functions and constraints with local coupling consensus. The most closely related approach to our algorithm is [16, Algorithm 4], with the following main difference: we propose a data server working at its own clock cycles that handles asynchronous data exchange among agents with no computation involved, while in [16] the authors use a centralised aggregator to take charge of data exchange, a part of the primal variable update and the dual variable update. We also introduce local consensus blocks instead of a common consensus, as well as a vectorised augmentation parameter instead of a scalar one.

The paper is organised as follows. Section II describes a distributed optimisation problem with local consensus constraints and a synchronous ADMM algorithm for its solution. Section III introduces the concept of a data exchange server in this context, explains the proposed asynchronous algorithm, and derives sufficient conditions on the problem and solver parameters for convergence. Section IV presents a numerical study illustrating the theoretical results of the paper and provides a comparison with an alternative approach. Concluding remarks and directions for future work are provided in Section V. Relevant proofs are included in the Appendix.

C. Notation

The $n \times n$ identity matrix and the n-dimensional column vector with all elements taking the value 1 are denoted by I_n and $\mathbf{1}_n$, respectively. A symmetric positive definite (or positive semidefinite) matrix is denoted $A \succ 0$ (or $A \succeq 0$, respectively). We define $\|x\|_Q^2 \overset{\text{def}}{=} x^\mathsf{T} Q x$ for $Q \succeq 0$. The indicator function of a nonempty closed convex set $\mathcal{C}$ is denoted $\mathcal{I}_\mathcal{C}(x)$, where $\mathcal{I}_\mathcal{C}(x) = 0$ for $x \in \mathcal{C}$ and $\mathcal{I}_\mathcal{C}(x) = +\infty$ otherwise. $\partial F(x)$ indicates the subdifferential of function F evaluated at x. Rounding to the nearest integer is denoted $\lfloor \cdot \rceil$. $N_{tr}(\mu, \sigma^2, a, b)$ indicates the truncated normal distribution[1].

II. Problem Statement

A. ADMM Formulation

ADMM considers the following convex optimisation problem:

$$\min_{x,y} \quad h_1(x) + h_2(y), \tag{1a}$$

$$\text{subject to} \quad Ax + By - c = 0. \tag{1b}$$

where $h_1 : \mathbb{R}^{n_1} \to \mathbb{R} \cup \{+\infty\}$, $h_2 : \mathbb{R}^{n_2} \to \mathbb{R} \cup \{+\infty\}$ are convex, closed and proper functions; A, B are matrices of appropriate dimension. We construct the augmented Lagrangian:

$$\mathcal{L}_\rho \overset{\text{def}}{=} h_1(x) + h_2(y) + \lambda^\mathsf{T}(Ax + By - c) + \frac{1}{2}\|Ax + By - c\|_\rho^2, \tag{2}$$

in which $\rho \succ 0$ is a penalty parameter.

In order to solve the problem, ADMM iteratively performs the following updates:

$$x \leftarrow \min_x \mathcal{L}_\rho, \tag{3a}$$

$$y \leftarrow \min_y \mathcal{L}_\rho, \tag{3b}$$

$$\lambda \leftarrow \lambda + \rho(Ax + By - c). \tag{3c}$$

ADMM guarantees [1] that if the problem (1) has a saddle point (x^*, y^*, λ^*), (i) the objective evaluated at the iterates of the primal variables (x, y) converge to its optimal value, (ii) the iterates of the primal residual $(Ax + By - c)$ converge to zero, and (iii) the iterates of the dual variable λ will converge to a saddle point.

B. Optimisation with Local Consensus

When the problems (3a) and (3b) are separable, they may be solved in a distributed manner. Here we propose a splitting scheme for distributed optimisation with local consensus. We consider a network of processing agents grouped by (i) $U \overset{\text{def}}{=} \{1, 2, 3 \dots M_U\}$ that solve separate problems in the form of (3a), and (ii) $V \overset{\text{def}}{=} \{1, 2, 3 \dots M_V\}$ that solve separate problems in the form of (3b). We assume (1b) has the special form of local coupling consensus constraints between the two groups, and use an undirected *bipartite graph* (bigraph) $\mathcal{G} = (U, V, E)$ to denote these relationships (see for example the illustration in Fig. 1). Thus the edge set E represents the local consensus couplings specified by the constraints (1b), which are formulated as the constraints (4c) below. We refer readers to [43] for a detailed description of modeling multi-agent networks using graph theory.

[1]If a random variable x has the normal distribution $N(\mu, \sigma^2)$ and $a < b$, then x conditional on $a \le x \le b$ follows $N_{tr}(\mu, \sigma^2, a, b)$. We specifically define $x \sim N_{tr}(\mu, \sigma^2, a, a)$ as $\mathbb{P}(x = a) = 1$.

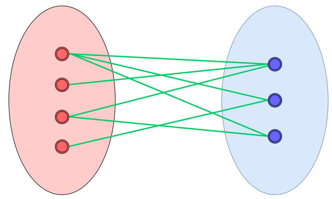

Fig. 1. Problem bigraph $\mathcal{G} = (U, V, E)$.

We reformulate (1) to obtain the main problem $\mathcal{P}$ considered in this paper:

$$\mathcal{P}: \min_{\substack{\{z_{ij}, w_{ij}\}_{(i,j)\in E},\\ \{u_i\}_{i\in U},\ \{v_j\}_{j\in V}}} \sum_{(i,j)\in E} \Big(F_{ij}(z_{ij}) + G_{ij}(w_{ij})\Big) \tag{4a}$$

$$+ \sum_{i\in U} f_i(u_i) + \sum_{j\in V} g_j(v_j), \tag{4b}$$

subject to:

$$z_{ij} = w_{ij}, \quad \forall (i,j)\in E, \tag{4c}$$

$$\big(u_i, \{z_{ij}\}_{j\in\mathcal{N}_i}\big) \in \mathcal{C}_i, \quad \forall i \in U, \tag{4d}$$

$$\big(v_j, \{w_{ij}\}_{i\in\mathcal{N}_j}\big) \in \mathcal{C}_j, \quad \forall j \in V, \tag{4e}$$

where $\mathcal{N}_i$ and $\mathcal{N}_j$ denote the sets of neighbours connected to agents i and j respectively. For each $i \in U$, $u_i \in \mathbb{R}^{p_i}$ is the local (private) decision variable and $z_{ij} \in \mathbb{R}^{m_{ij}}$ is the local consensus decision variable to be shared with $j \in \mathcal{N}_i$. Similarly, for each $j \in V$, $v_j \in \mathbb{R}^{p_j}$ is the local (private) decision variable and $w_{ij} \in \mathbb{R}^{m_{ij}}$ is to be shared with $i \in \mathcal{N}_j$. Constraint sets $\mathcal{C}_i$ and $\mathcal{C}_j$ represent inequality constraints that apply to u_i and v_i and their local consensus variables $\{z_{ij}\}_{j\in\mathcal{N}_i}$ and $\{w_{ij}\}_{i\in\mathcal{N}_j}$, respectively. Since the graph $\mathcal{G}$ is undirected, $j \in \mathcal{N}_i$ if and only if $i \in \mathcal{N}_j$.

Assumption 1. *Problem $\mathcal{P}$ has the following properties:*

(a). $\{F_{ij}, G_{ij} : \mathbb{R}^{m_{ij}} \to \mathbb{R}\}_{(i,j)\in E}$, $\{f_i : \mathbb{R}^{p_i} \to \mathbb{R}\}_{i\in U}$, *and* $\{g_j : \mathbb{R}^{p_j} \to \mathbb{R}\}_{j\in V}$ *are convex objectives.*

(b). *$\mathcal{C}_i$ and $\mathcal{C}_j$ are convex local inequality constraint sets.*

Remark 1. *Several applications have the same structure as problem $\mathcal{P}$. For example, supply-demand pairs [44] in the scenario of market behavior and individual-regulator pairs [45] in the scenario of resource allocation.*

Problem $\mathcal{P}$ in (4) is equivalent to (1) under the assignments:

- $h_1 = \sum_{i\in U} \big(\sum_{j\in\mathcal{N}_i} F_{ij}(z_{ij}) + f_i(u_i) + \mathcal{I}_{\mathcal{C}_i}(u_i, \{z_{ij}\}_{j\in\mathcal{N}_i})\big)$
- $h_2 = \sum_{j\in V} \big(\sum_{i\in\mathcal{N}_j} G_{ij}(w_{ij}) + g_j(v_j) + \mathcal{I}_{\mathcal{C}_j}(v_i, \{w_{ij}\}_{i\in\mathcal{N}_j})\big)$
- (1b) is equivalent to (4c).

A local sub-problem $\mathcal{P}_i$ is defined for each $i \in U$ as

$$\mathcal{P}_i: \min_{u_i, \{z_{ij}\}_{j\in\mathcal{N}_i}} f_i(u_i) + \sum_{j\in\mathcal{N}_i} F_{ij}(z_{ij}), \tag{5a}$$

$$\text{subject to: } \big(u_i, \{z_{ij}\}_{j\in\mathcal{N}_i}\big) \in \mathcal{C}_i, \tag{5b}$$

and likewise $\mathcal{P}_j$ is defined for each agent $j \in V$ as

$$\mathcal{P}_j: \min_{v_j, \{w_{ij}\}_{i\in\mathcal{N}_j}} g_j(v_j) + \sum_{i\in\mathcal{N}_j} G_{ij}(w_{ij}), \tag{6a}$$

$$\text{subject to: } \big(v_j, \{w_{ij}\}_{i\in\mathcal{N}_j}\big) \in \mathcal{C}_j. \tag{6b}$$

By dualising the coupling constraints (4c), we obtain the augmented Lagrangian of problem $\mathcal{P}$:

$$\begin{aligned}\mathcal{L}_\Theta \overset{\text{def}}{=} &\sum_{i\in U} \Big(f_i(u_i) + \sum_{j\in\mathcal{N}_i} F_{ij}(z_{ij})\Big) \\ &+ \sum_{j\in V} \Big(g_j(v_j) + \sum_{i\in\mathcal{N}_j} G_{ij}(w_{ij})\Big) \\ &+ \sum_{(i,j)\in E} \Big(\lambda_{ij}^\intercal (z_{ij} - w_{ij}) + \frac{1}{2}\|z_{ij} - w_{ij}\|^2_{\Theta_{ij}}\Big).\end{aligned} \tag{7}$$

Here $\{\Theta_{ij}\}_{(i,j)\in E}$ is a set of penalty parameters that control the step size, and hence the convergence rate, of the Method of Multipliers applied to $\mathcal{P}$ (see e.g. [1] Sec. 3). Conditions on $\{\Theta_{ij}\}_{(i,j)\in E}$ to ensure convergence of the proposed asynchronous ADMM are identified in Section III and investigated numerically in Section IV. In this section we simply make the following assumption.

Assumption 2. $\Theta_{ij} \succ 0,\ \forall (i,j) \in E$.

We define $\mathcal{L}^i_\Theta, \forall i \in U$, and $\mathcal{L}^j_\Theta, \forall j \in V$ as:

$$\begin{aligned}\mathcal{L}^i_\Theta \overset{\text{def}}{=} f_i(u_i) + &\sum_{j\in\mathcal{N}_i} \Big(F_{ij}(z_{ij}) \\ &+ \lambda_{ij}^\intercal (z_{ij} - w_{ij}) + \frac{1}{2}\|z_{ij} - w_{ij}\|^2_{\Theta_{ij}}\Big),\end{aligned} \tag{8}$$

$$\begin{aligned}\mathcal{L}^j_\Theta \overset{\text{def}}{=} g_j(v_j) + &\sum_{i\in\mathcal{N}_j} \Big(G_{ij}(w_{ij}) \\ &+ \lambda_{ij}^\intercal (z_{ij} - w_{ij}) + \frac{1}{2}\|z_{ij} - w_{ij}\|^2_{\Theta_{ij}}\Big).\end{aligned} \tag{9}$$

Applying synchronous ADMM (3) to this problem results in Algorithm 1. Each iteration of this algorithm involves the following steps:

- In Step 1, each agent $i \in U$ solves problem $\mathcal{P}_i$ using $\{w_{ij}, \lambda_{ij}\}_{i\in\mathcal{N}_j}$ computed at the previous iteration and, for each $j \in \mathcal{N}_i$, sends the updated local consensus variable z_{ij} to agent j.
- Similarly, in Step 2, each agent $j \in V$ solves problem $\mathcal{P}_j$ using $\{z_{ij}, \lambda_{ij}\}_{i\in\mathcal{N}_j}$ computed at the previous iteration and sends the updated w_{ij} to agent i, for each $i \in \mathcal{N}_j$.
- In Step 3, all agents cooperatively update the Lagrange multipliers $\{\lambda_{ij}\}_{(i,j)\in E}$; hence the local iterates λ_{ij} of i and j are identical for all $(i,j) \in E$ at each iteration.

Assumption 3. *Assume that:*

(a). *The Lagrangian (7) has at least one saddle point* $\{u_i^*\}_{i\in U}$, $\{v_j^*\}_{j\in V}$, $\{z_{ij}^*, w_{ij}^*, \lambda_{ij}^*\}_{(i,j)\in E}$.

(b). *All the U and V updates in Algorithm 1 have solutions for any inputs.*

Remark 2. *Assumption 3(b) is easily achieved since, for all $\forall i \in U$, $\mathcal{L}^i_\Theta$ is strongly convex in z_{ij} under Assumption 2, and the same reasoning applies to $\mathcal{L}^j_\Theta$, $\forall j \in V$.*

Algorithm 1 Solve $\mathcal{P}$ via Synchronous ADMM

Initialise: z_{ij}, w_{ij}, λ_{ij}, Θ_{ij} $(\forall (i,j) \in E)$

Repeat:

1: U Update $(\forall i \in U$ in parallel$)$

Input: $\{w_{ij}, \lambda_{ij}\}_{j \in \mathcal{N}_i}$

Output:

$$u_i, \{z_{ij}\}_{j \in \mathcal{N}_i} \leftarrow \underset{u_i, \{z_{ij}\}_{j \in \mathcal{N}_i}}{\arg\min} \ \mathcal{L}^i_\Theta\big(u_i, \{z_{ij}, w_{ij}, \lambda_{ij}\}_{j \in \mathcal{N}_i}\big) \quad \text{s.t.} \quad \big(u_i, \{z_{ij}\}_{j \in \mathcal{N}_i}\big) \in \mathcal{C}_i$$

Each $j \in \mathcal{N}_i$ communicates $\{z_{ij}\}_{j \in \mathcal{N}_i}$ to the respective j

2: V Update $(\forall j \in V$ in parallel$)$

Input: $\{z_{ij}, \lambda_{ij}\}_{j \in \mathcal{N}_i}$

Output:

$$v_j, \{w_{ij}\}_{i \in \mathcal{N}_j} \leftarrow \underset{v_j, \{w_{ij}\}_{i \in \mathcal{N}_j}}{\arg\min} \ \mathcal{L}^j_\Theta\big(v_j, \{w_{ij}, z_{ij}, \lambda_{ij}\}_{i \in \mathcal{N}_j}\big) \quad \text{s.t.} \quad \big(v_j, \{w_{ij}\}_{i \in \mathcal{N}_j}\big) \in \mathcal{C}_j$$

Each $i \in \mathcal{N}_j$ communicates $\{w_{ij}\}_{i \in \mathcal{N}_j}$ to the respective i

3: E Update $(\forall i \in U$ and $\forall j \in V$ in parallel$)$

Input: $\{z_{ij}, w_{ij}, \lambda_{ij}\}_{(i,j) \in E}$

$$\lambda_{ij} \leftarrow \lambda_{ij} + \Theta_{ij}(z_{ij} - w_{ij})$$

Until satisfaction of the stopping criterion

Output:

4:

$$u_i, \{z_{ij}\}_{j \in \mathcal{N}_i} \quad \forall i \in U$$
$$v_j, \{w_{ij}\}_{i \in \mathcal{N}_j} \quad \forall j \in V$$

Theorem 1. *Under Assumptions 1, 2, and 3, the iterates* u_i, $\{z_{ij}, \lambda_{ij}\}_{j \in \mathcal{N}_i}$, $\forall i \in U$, *and* v_j, $\{w_{ij}, \lambda_{ij}\}_{i \in \mathcal{N}_j}$, $\forall j \in V$ *of Algorithm 1 have the following convergence properties:*

1) *the objective of* $\mathcal{P}$ *evaluated at the local iterates converges to the optimal value;*
2) *the primal residual* $\sum_{\forall (i,j) \in E}(z_{ij} - w_{ij})$ *evaluated at the local iterates converges to zero.*

Remark 3. *Algorithm 1 is required to synchronise communication between agents twice within each ADMM iteration. Therefore any unreliable peer-to-peer connections* $(i,j) \in E$ *will increase the waiting time needed per iteration. To avoid this problem and allow more flexible inter-agent communications, an asynchronous ADMM algorithm with a data exchange server is proposed in the following section.*

III. Network Model and Proposed Asynchronous ADMM Algorithm

We introduce a *Data Exchange Server* to handle the shared data among the participating agents. Each agent is directly connected to the server via a communication link with a different round-trip time, as illustrated in Fig. 2.

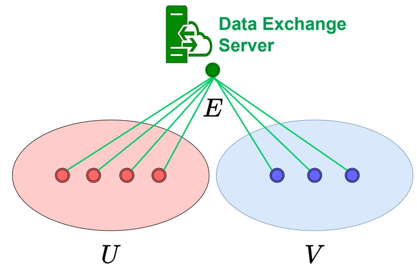

Fig. 2. Network graph with a data exchange server.

Algorithm 2 Decentralised Asynchronous ADMM - (1/3) Data Exchange Server

Initialise: z_{ij}, w_{ij}, Θ_{ij} $\forall (i,j) \in E$

$k = k_0 \le -\max(\{\tau_i\} \cup \{\tau_j\})$

1: Send the initial data:

$\{\{w^{\text{ini}}_{ij}\}_{j \in \mathcal{N}_i}\} \ \to \forall i \in U$

$\{\{z^{\text{ini}}_{ij}\}_{i \in \mathcal{N}_j}\} \ \to \forall j \in V$

Repeat:

2: During clock cycle k:

Receive data from inbound agents:

$\{z^k_{ij}\}_{j \in \mathcal{N}_i} \leftarrow \{z^{\text{in}}_{ij}\}_{j \in \mathcal{N}_i}$ $\quad \forall i \in U$ s.t. $a^1_i(k) = k$

$\{w^k_{ij}\}_{i \in \mathcal{N}_j} \leftarrow \{w^{\text{in}}_{ij}\}_{i \in \mathcal{N}_j}$ $\quad \forall j \in V$ s.t. $b^1_j(k) = k$

For non-inbound agents:

$\{z^k_{ij}\}_{j \in \mathcal{N}_i} \leftarrow \{z^{k-1}_{ij}\}_{j \in \mathcal{N}_i}$ $\quad \forall i \in U$ s.t. $a^1_i(k) < k$

$\{w^k_{ij}\}_{i \in \mathcal{N}_j} \leftarrow \{w^{k-1}_{ij}\}_{i \in \mathcal{N}_j}$ $\quad \forall j \in V$ s.t. $b^1_j(k) < k$

3: At the end of clock cycle k:

Respond to the inbound agents with the data:

$\{\{w^l_{ij}\}_{j \in \mathcal{N}_i}\}^k_{l=a^2_i(k)+1} \to i$ $\quad \forall i \in U$ s.t. $a^1_i(k) = k$

$\{\{z^l_{ij}\}_{i \in \mathcal{N}_j}\}^k_{l=b^2_j(k)+1} \to j$ $\quad \forall j \in V$ s.t. $b^1_j(k) = k$

4: $k \leftarrow k + 1$

Until $k = K$, send terminating signal to all agents.

The clock cycles of the data server are indexed by $k \in \{k_0, k_0+1, \ldots, -1, 0, 1, \ldots, K\}$, where $k_0 < 0$. During each clock cycle the server receives the data from an arbitrary set of agents. At the end of the clock cycle, the data server sends to each agent from which it received data in that cycle a set of data that it has received from the respective coupling agents. We refer the reader to Algorithm 2 for the details of how the data exchange server operates. To understand this algorithm:

- As shown in Fig. 3, $a^1_i(k)$ denotes the most recent clock cycle before the end of cycle k in which data from agent $i \in U$ arrived at the server, and $a^2_i(k)$ the next most recent one. Similarly, $b^1_j(k)$ denotes the most recent cycle before the end of cycle k in which data from agent $j \in V$ arrived at the server, and $b^2_j(k)$ the next most recent one. During the first few cycles when $a^2_i(k)$ and $b^2_j(k)$ are not defined, $a^2_i(k)$ and $b^2_j(k)$ are set to k_0, the initial clock cycle.
- The clock cycle counter is initialised as $k = k_0 \le -\max(\{\tau_i\} \cup \{\tau_j\})$, where τ_i and τ_j are defined in Assumption 4 and represent available bounds on com-

munication delays. This choice ensures that all variables have been updated at least once before $k = 1$ (see Algorithm 5) and also allows the window of a running average output to be adjusted by tuning k_0 and K.

- The algorithm starts when the initial data is sent to the respective agents in Step 1.
- During each clock cycle k, the server passively collects the consensus updates from the agents as shown in Step 2. The server receives consensus updates $\{z_{ij}^{\text{in}}\}, \{w_{ij}^{\text{in}}\}$ from inbound agents, namely $i \in U$ such that $a_i^1(k) = k$ and $j \in V$ such that $b_j^1(k) = k$. These saved as the recorded updates $\{z_{ij}^k\}, \{w_{ij}^k\}$. For the rest of the (non-inbound) agents whose consensus updates have not arrived at the server during clock cycle k, represented by $a_i^1(k) < k$ or $b_j^1(k) < k$, the server saves duplicates of the consensus data of the previous cycle as the recorded updates.
- At the end of clock cycle k, the server responds to all the inbound agents with all the historical recorded updates of their counterparts, to which they are connected by the respective edge in E, since the last communication, as shown in Step 3.

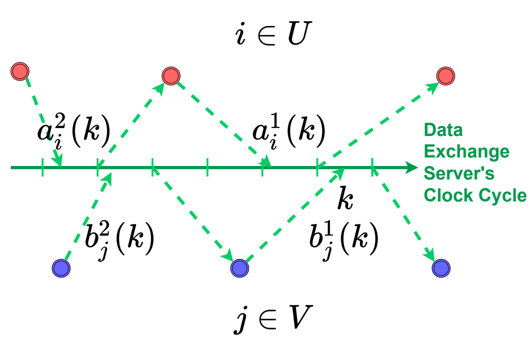

Fig. 3. Clock cycles of the data exchange server.

Every agent works responsively and asynchronously. When it receives data from the server, the agent computes the update and replies to the server according to Algorithm 3 or 4. In particular:

- In Step 1, the agent passively receives from the server a time sequence of recorded consensus updates from its connected counterparts, which was sent in Step 3 in Algorithm 2.
- The agent then reconstructs the λ_{ij} by adding the primal residuals as shown in Step 2. Note that for $\forall i \in U$ and $\forall j \in V$ there is a slight difference in such additions.
- In Step 3, the agent updates in the same way as Step 1 or 2 in Algorithm 1.
- The agent responds to the server with the updated consensus data in Step 4.
- In Step 5, the agent records the weights of its historical iterates, making preparations for the running-average output in Step 6.
- When the algorithm is terminated, the agent outputs the weighted running average as in Step 6. The window sizes C_i and C_j of the running average output are adjustable

Algorithm 3 Decentralised Asynchronous ADMM - (2/3) $\forall i \in U$ in parallel

Repeat:

1: Receive data from server: $\{\{w_{ij}^l\}_{j\in\mathcal{N}_i}\}_{l=1}^L$

2: $$\lambda_{ij} \leftarrow \lambda_{ij} + \sum_{l=1}^{L-1} \Theta_{ij}(z_{ij}^{c-1} - w_{ij}^l) + \Theta_{ij}(z_{ij}^c - w_{ij}^L) \qquad \forall j \in \mathcal{N}_i$$

3: $$u_i, \{z_{ij}\}_{j\in\mathcal{N}_i} \leftarrow \operatorname*{arg\,min}_{u_i, \{z_{ij}\}_{j\in\mathcal{N}_i}} \mathcal{L}_\Theta^i\big(u_i, \{z_{ij}, w_{ij}^L, \lambda_{ij}\}_{j\in\mathcal{N}_i}\big) \quad \text{s.t. } \big(u_i, \{z_{ij}\}_{j\in\mathcal{N}_i}\big) \in \mathcal{C}_i$$

4: Send data to server: $\{z_{ij}\}_{j\in\mathcal{N}_i}$

5: $$d^c \leftarrow L \overset{\text{def}}{=} \text{length}(\{\{w_{ij}\}_{j\in\mathcal{N}_i}\})$$
$$c \leftarrow c+1$$
$$u_i^c, \{z_{ij}^c\} \leftarrow u_i, \{z_{ij}\}_{j\in\mathcal{N}_i}$$

Until receive the terminating signal from the server.

Output:

6: $$\bar{u}_i, \{\bar{z}_{ij}\}_{j\in\mathcal{N}_i} \overset{\text{def}}{=} \frac{\sum_{c\geq C_i} d^c\{u_i^c, \{z_{ij}^c\}\}}{\sum_{c\geq C_i} d^c} \tag{10}$$

and could be either set to k_0 or set independently (these choices being equivalent in the limit as $K \to \infty$).

Algorithm 5 provides a summary of Algorithms 2, 3 and 4, after simplification by removing the detailed description of the information that passes through the data exchange server. To understand this:

- Steps 1 and 2 of Algorithm 5 resemble Steps 1 and 2 of the synchronous Algorithm 1, but with historical data.
- The local reconstructions of λ_{ij} (in Step 2 of Algorithms 3 and 4) are equivalent to Step 3 of Algorithm 5.
- The weighted running averages in Step 6 of Algorithms 4 and 5 are equivalent to the arithmetic average in Step 5 of Algorithm 5 since duplicates are recorded in clock cycles in which no data is received by the data exchange server.

Assumption 4. *We assume the following conditions:*

(a). *Bounded delay and postponed conflicts* $\forall k$*:*

$$1 \leq a_i^1(k) - a_i^2(k) \leq \tau_i, \quad \forall i \in U, \tag{15}$$
$$1 \leq b_j^1(k) - b_j^2(k) \leq \tau_j, \quad \forall j \in V. \tag{16}$$

(b). *For all* $(i,j) \in E$ *with* $\tau_i \neq 1$*,* F_{ij} *is strongly convex with a generalised modulus* $\Sigma_{ij}^U \succ 0$ *defined as:*

$$\begin{aligned} &\partial F_{ij}(z_{ij}^\dagger)^\intercal(z_{ij} - z_{ij}^\dagger) + \frac{1}{2}\|z_{ij} - z_{ij}^\dagger\|_{\Sigma_{ij}^U}^2 \\ \leq &F_{ij}(z_{ij}) - F_{ij}(z_{ij}^\dagger), \quad \forall z_{ij}^\dagger, z_{ij} \in \mathbb{R}^{m_{ij}}. \end{aligned} \tag{17}$$

(c). *For all* $(i,j) \in E$*,* G_{ij} *is strongly convex with a gener-*

Algorithm 4 Decentralised Asynchronous ADMM - (3/3) $\forall j \in V$ in parallel

Repeat:

Receive data from server: $\{\{z_{ij}^l\}_{i\in\mathcal{N}_j}\}_{l=1}^L$

$\lambda_{ij} \leftarrow \lambda_{ij} + \sum_{l=1}^L \Theta_{ij}(z_{ij}^l - w_{ij}^{c-1})$ $\quad \forall i \in \mathcal{N}_j$

$v_j, \{w_{ij}\}_{i\in\mathcal{N}_j} \leftarrow$

$$\underset{v_j,\{w_{ij}\}_{i\in\mathcal{N}_j}}{\arg\min}\ \mathcal{L}_\Theta^j\big(v_j, \{w_{ij}, z_{ij}^L, \lambda_{ij}\}_{i\in\mathcal{N}_j}\big)$$

$$\text{s.t. } \big(v_j, \{w_{ij}\}_{i\in\mathcal{N}_j}\big) \in \mathcal{C}_j$$

Send data to server: $\{w_{ij}\}_{i\in\mathcal{N}_j}$

$d^c \leftarrow L \stackrel{\text{def}}{=} \text{length}(\{\{z_{ij}\}_{i\in\mathcal{N}_j}\})$

$c \leftarrow c+1$

$v_j^c, \{w_{ij}^c\} \leftarrow v_j, \{w_{ij}\}_{i\in\mathcal{N}_j}$

Until receive the terminating signal from the server.

Output:

$$\bar{v}_j, \{\bar{w}_{ij}\}_{i\in\mathcal{N}_j} \stackrel{\text{def}}{=} \frac{\sum_{c\geq C_j} d^c\{v_j^c, \{w_{ij}^c\}\}}{\sum_{c\geq C_j} d^c} \tag{11}$$

alised modulus $\Sigma_{ij}^V \succ 0$ defined as:

$$\begin{aligned}&\partial G_{ij}(w_{ij}^\dagger)^\intercal(w_{ij} - w_{ij}^\dagger) + \frac{1}{2}\|w_{ij} - w_{ij}^\dagger\|^2_{\Sigma_{ij}^V} \\ &\leq G_{ij}(w_{ij}) - G_{ij}(w_{ij}^\dagger), \quad \forall w_{ij}^\dagger, w_{ij} \in \mathbb{R}^{m_{ij}}.\end{aligned} \tag{18}$$

$\forall(i,j) \in E$, we define τ_{ij} and α_{ij} as

$$\tau_{ij} \stackrel{def}{=} 2\tau_i + 2\tau_j - 4, \tag{19}$$

$$\alpha_{ij} \stackrel{def}{=} 1 + \frac{1}{2}(3\tau_{ij} + \sqrt{5\tau_{ij}^2 + 8\tau_{ij} + 4}). \tag{20}$$

(d). *$\forall(i,j) \in E$ such that $\tau_i \neq 1$:*

$$\frac{\Sigma_{ij}^U}{\alpha_{ij}(4\tau_i - 4)} - \Theta_{ij} \succeq 0. \tag{21}$$

(e). *$\forall(i,j) \in E$:*

$$\frac{\Sigma_{ij}^V}{\alpha_{ij}(4\tau_j - 3)} - \Theta_{ij} \succeq 0. \tag{22}$$

The convergence of the proposed asynchronous ADMM can be stated as follows (a proof is provided in the Appendix).

Theorem 2. *Let Assumptions 1, 3 and 4 hold. Then Algorithms 2, 3 and 4 (or equivalently Algorithm 5 in the limit as $K \to \infty$) have the following asymptotic properties:*

1) *The reconstructed local running averages $\bar{u}_i$, $\{\bar{z}_{ij}\}_{j\in\mathcal{N}_i}$, $\forall i \in U$ in Algorithm 3 and $\bar{v}_j$, $\{\bar{w}_{ij}\}_{i\in\mathcal{N}_j}$, $\forall j \in V$ in Algorithm 4 converge as $K \to \infty$ to a saddle point $\{u_i^*\}, \{v_j^*\}, \{z_{ij}^*, w_{ij}^*\}_{(i,j)\in E}$ of the Lagrangian (7).*

2) *Equivalently, $\{\bar{u}_i^K\}$, $\{\bar{v}_j^K\}$, $\{\bar{z}_{ij}^K, \bar{w}_{ij}^K\}_{(i,j)\in E}$ in Algorithm 5 converge to $\{u_i^*\}$, $\{v_j^*\}$, $\{z_{ij}^*, w_{ij}^*\}_{(i,j)\in E}$ as $K \to \infty$.*

Algorithm 5 Decentralised Asynchronous ADMM - Complete Picture

Initialise: z_{ij}, w_{ij}, λ_{ij}, Θ_{ij} $\quad \forall (i,j) \in E$

Repeat:

$\forall i \in U$, $\forall j \in V$ at server's clock cycle $k \geq 1$:

U Update $\quad \forall i \in U$

$u_i^k, \{z_{ij}^k\}_{j\in\mathcal{N}_i} \leftarrow$

$$\underset{u_i,\{z_{ij}\}_{j\in\mathcal{N}_i}}{\arg\min}\ \mathcal{L}_\Theta^i\big(u_i, \{z_{ij}, w_{ij}^{a_i^2(k)}, \lambda_{ij}^{a_i^2(k)}\}_{j\in\mathcal{N}_i}\big) \tag{12}$$

$$\text{s.t.} \quad \big(u_i, \{z_{ij}\}_{j\in\mathcal{N}_i}\big) \in \mathcal{C}_i$$

V Update $\quad \forall j \in V$

$v_j^k, \{w_{ij}^k\}_{i\in\mathcal{N}_j} \leftarrow$

$$\underset{v_j,\{w_{ij}\}_{i\in\mathcal{N}_j}}{\arg\min}\ \mathcal{L}_\Theta^j\big(v_j, \{w_{ij}, z_{ij}^{b_j^2(k)}, \lambda_{ij}^{b_j^2(k)-1}\}_{i\in\mathcal{N}_j}\big) \tag{13}$$

$$\text{s.t. } \big(v_j, \{w_{ij}\}_{i\in\mathcal{N}_j}\big) \in \mathcal{C}_j$$

The local reconstruction of λ_{ij} is equivalent to:

E Update $\quad \forall (i,j) \in E$, at server's clock cycle k

$$\lambda_{ij}^k \leftarrow \lambda_{ij}^{k-1} + \Theta_{ij}(z_{ij}^k - w_{ij}^k) \tag{14}$$

$k \leftarrow k+1$

Until $k = K$

Output:

$\bar{u}_i^K, \{\bar{z}_{ij}^K\}_{j\in\mathcal{N}_i} \quad \forall i \in U$

$\bar{v}_j^K, \{\bar{w}_{ij}^K\}_{i\in\mathcal{N}_j} \quad \forall j \in V$

in which the running average $\bar{x}^K \stackrel{\text{def}}{=} \frac{1}{K}\sum_{k=1}^K x^k$

IV. Numerical Analysis and Comparison

This section investigates the convergence properties of the proposed algorithm through numerical simulations. The example considered is the following modified Ridge regression problem (linear regression with ℓ_2 regularisation):

$$\begin{aligned}\min_{\{z_i\}_{i\in U}} \quad & \sum_{i\in U} r_i\|z_i\|_2^2 \\ & + \sum_{j\in V}\Big(\sum_{i\in U}\big(\|A_{ij}z_i - b_{ij}\|_2^2 + \sum_{k\in U} c_j\|z_i - z_k\|_2^2\big)\Big)\end{aligned} \tag{23a}$$

subject to

$$\text{s.t. } \underline{z}\mathbf{1}_n \leq z_i \leq \bar{z}\mathbf{1}_n, \ \forall i \in U. \tag{23b}$$

We assume $z_i \in \mathbb{R}^n$, $r_i > 0$, $\forall i \in U$; $c_j > 0$, $\forall j \in V$; $A_{ij} \in \mathbb{R}^{m\times n}$, $b_{ij} \in \mathbb{R}^m$, $\forall (i,j) \in U \times V$. This problem can be viewed as $|U|$ independent learning nodes that identify their respective parameters $\{z_i\}$ via the local data $\{A_{ij},\ b_{ij}\}$

stored in the $|V|$ data centres, with $\{r_i\}$ being the penalty terms for ℓ_2 regularisation. Data centres may have prior knowledge that some parameters are related, and this motivates the inclusion of the penalty terms $\{c_j\}$. The vectors containing the elements of the matrices A_{ij} are each drawn from the normal distribution $N(0, I_{mn})$; each b_{ij} is generated using $b_{ij} = A_{ij}\hat{z}_i + d_{ij}$, where the noise vector d_{ij} is drawn from $N(0, 0.01 I_m)$, and $\hat{z}_i = z^{\text{ref}} + e_i$ where each element of z^{ref} is zero with probability 0.5 and otherwise is drawn from $N(0,1)$, and the noise e_i is drawn from $N(0, 0.01 I_n)$. The remaining coefficients are $|U| = 4$, $|V| = 4$, $n = 10$, $\underline{z} = -2$, $\bar{z} = 2$, $\{c_j = 10\}_{\forall j \in V}$. We reformulate Problem (23) equivalently as

$$\min_{\{z_i\}_{i\in U}, \{w_{ij}\}_{(i,j)\in U\times V}} \sum_{i\in U} r_i \|z_i\|_2^2 \tag{24a}$$

$$+ \sum_{j\in V} \Big(\sum_{i\in U} (\|A_{ij} w_{ij} - b_{ij}\|_2^2 + \sum_{k\in U} c_j \|w_{ij} - w_{kj}\|_2^2) \Big) \tag{24b}$$

subject to:

$$\underline{z}\mathbf{1}_n \le z_i \le \bar{z}\mathbf{1}_n, \ \forall i \in U, \tag{24c}$$

$$z_i = w_{ij}, \ \forall (i,j) \in U \times V. \tag{24d}$$

To see the equivalence of this with problem $\mathcal{P}$, note that each $i \in U$ has the decision variable z_i with the local cost function (24a) and the local constraint set (24c), whereas each $j \in V$ has local decision variables $\{w_{ij}\}_{\forall i\in U}$ with the local objective (24b). The realisation of the delay t_i, $i \in U$, is modelled as: $t_i \sim \lfloor N_{tr}(\frac{\tau_i+1}{2}, (\frac{\tau_i-1}{4})^2, 1, \tau_i) \rceil$, and t_j, $j \in V$, is modelled analogously. The delay upper bounds are identical for all $i \in U$ or $j \in V$, namely $\{\tau_i = \tau_U\}_{\forall i\in U}$ and $\{\tau_j = \tau_V\}_{\forall j \in V}$.

Fig. 4 displays the convergence behaviour of the proposed asynchronous ADMM algorithm. We define the residual of the objective value $\text{R}^{\text{obj}}(k) \overset{\text{def}}{=} \frac{|\text{obj}^k - \text{obj}^*|}{|\text{obj}^*|}$ where obj^* is the optimal objective value obtained with a centralised solver. The parameter vector $\text{p}_s \overset{\text{def}}{=} [\theta, \tau_U, \tau_V]$ summarises the simulation parameters, where θ is a scalar defining $\{\Theta_{ij} = \theta I_n\}_{\forall (i,j)\in E}$. We also compute $\theta_r \overset{\text{def}}{=} [\hat{\theta}, \bar{\theta}]$, in which $\hat{\theta}$ is the step size computed using (21) and (22) in Assumption 4, evaluated using the upper bounds on delays, and $\bar{\theta}$ is the corresponding step size evaluated at the expected values of the delays.

When the local objective functions of all the agents are strongly convex, we observe from Fig. 4(a) that the iterations converge until θ has increased to a critical value, above which the iterations diverge. In this specific example, the threshold from the simulation is a factor of 10^3 higher than $\hat{\theta}$ and 10^2 higher than $\bar{\theta}$. If the local objectives of the agents in U are not strongly convex, the simulation result in Fig. 4(b) shows that (i) the critical value of θ increases as the maximum delays decrease; (ii) when we interchange the value of τ_U and τ_V, the case that $\tau_U > \tau_V$ results in larger critical value of θ, even when θ_r diminishes to zero due to the loss of strong convexity; (iii) when $\tau_U = \tau_V = 1$, the iteration converges even when θ is increased to 10^4. Fig. 4(c) presents the results when local objectives of V are not strongly convex. We observe: (i) when τ_V is greater than 1, the iterations diverge no matter how small θ is; (ii) Lower τ_U implies higher critical value for θ; (iii) when $\tau_U = \tau_V = 1$, similar to (b)(iii), the iteration converges at high θ. To summarise the numerical analysis: the processing agents in V are more intolerant both to non-strongly-convex local objectives and larger delays (that require lower θ values for convergence). These observations are consistent with Theorem 2 but they also indicate that the sufficient conditions provided by (21) and (22) in Assumption 4 for the critical value of θ are conservative.

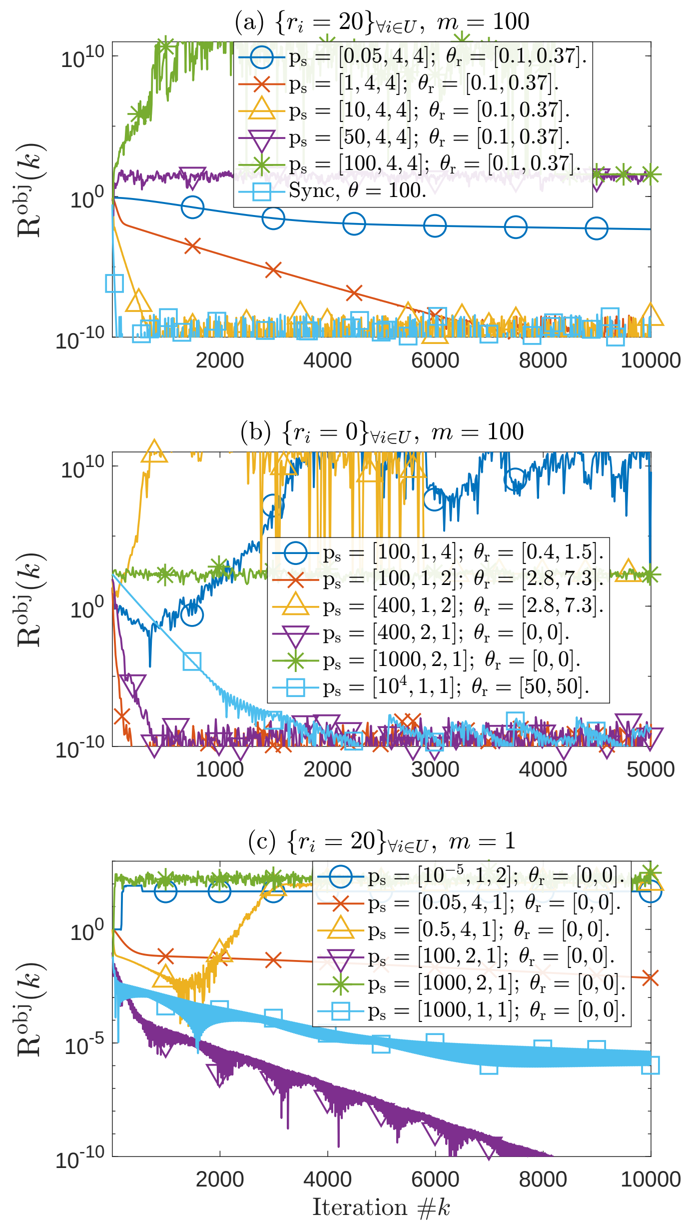

Fig. 4. Convergence with the local objective functions of the agents (a) all being strongly convex, (b) only being strongly convex in group $\boldsymbol{V}$, (c) only being strongly convex in group $\boldsymbol{U}$.

Problem (23) was further explored using the distributed ADMM approach of [16, Algorithm 4]. In this approach, a computing aggregator (see Fig. 5) replaces the data exchange server previously shown in Fig. 2. This aggregator is configured to maintain local copies of all the consensus decision variables $\{z_i\}_{\forall i\in U}, \{w_{ij}\}_{\forall (i,j)\in U\times V}$, executing updates in concordance with the form in [1, Sec. 7.2]. Owing

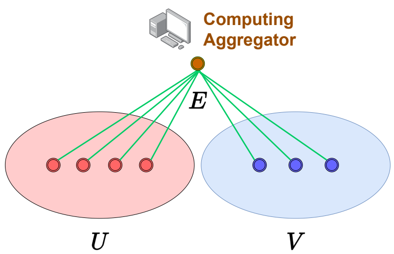

Fig. 5. Replacing the data exchange server with an aggregator.

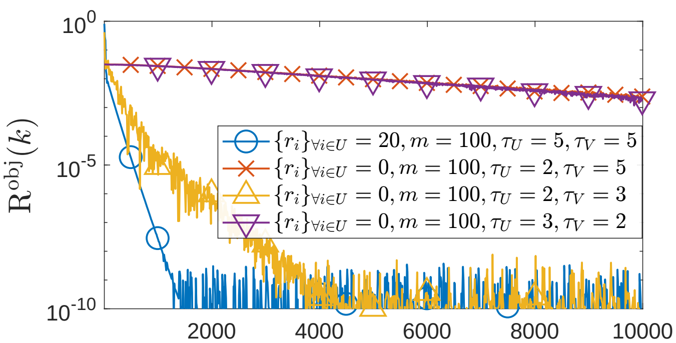

Fig. 6. Convergence when $\boldsymbol{\theta} \to \boldsymbol{\theta}^{\mathbf{lim}}$ with [16, Algorithm 4].

to the substantial alterations made to the ADMM structure, the convergence rates of the two algorithms are not directly comparable. However Fig. 6 provides an indication of the performance of [16, Algorithm 4] for several different delay bounds when its penalty parameter θ is set to the empirical upper limit θ^{lim} that has the highest convergence rate.

The plots in Fig. 6 are analogous to those in Fig. 4(a) (for $\{r_i\}_{\forall i \in U} = 20$) and Fig.4 (for $\{r_i\}_{\forall i \in U} = 0$). In this comparison, an extra clock cycle must be included in the delay for [16, Algorithm 4] due to the change from a data exchange server to a computing aggregator, as depicted in Fig. 7. From qualitative comparison of Fig. 6 with Fig. 4, we conclude that the proposed asynchronous ADMM converges rapidly when applied to the problems for which [16, Algorithm 4] converges within 5000 iterations, and moreover the proposed algorithm also converges in one of the two cases shown in Fig. 6 in which the convergence of [16, Algorithm 4] is impractically slow. Since it does not require a communication system with OSI (Open Systems Interconnection [46]) Layer 6 (Presentation) and Layer 7 (Application), the proposed method using a computation-free server considerably reduces processing time, resulting in smaller delays and/or allowing faster clock cycles. This characteristic potentially facilitates the server's integration into existing communication infrastructure.

Moreover, the absence of an encryption/decryption process in the proposed data exchange server inherently safeguards peer-to-peer privacy, thereby strengthening its appeal as a viable alternative to a computing aggregator. The data exchange server necessitates larger memory allocation to cache historical data. Quantitatively, this is approximately p times the amount employed by the aggregator, where p is proportional to the average of $\{\frac{1}{2}(\frac{\tau_i}{\tau_j} + \frac{\tau_j}{\tau_i})\}_{(i,j) \in E}$. However, it is crucial to recognise that when computational aspects are taken into account, the aggregator's memory requirements may significantly exceed those of the data exchange server. Regarding the communication cost, the exchange server maintains a data transfer rate equivalent to that of the synchronous case (albeit with fluctuations due to asynchrony), and when compared with a computing aggregator, it offers bandwidth savings by eliminating the need to transfer data for dual variable updates.

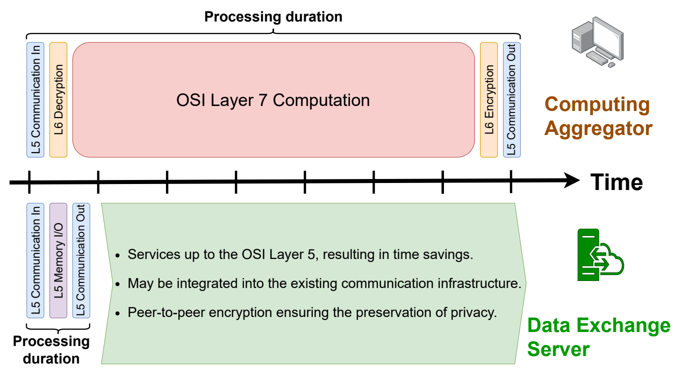

Fig. 7. Key differences between a computing aggregator and a data exchange server.

V. Conclusion

This paper proposes an asynchronous, distributed ADMM optimisation algorithm for problems with local consensus coupling constraints, in which a computation-free data exchange server handles the communication between agents with delays. Under assumptions of strongly convex local objectives and upper limits on communication delays, sufficient conditions are derived on the penalty parameters in the augmented Lagrangian formulation in order to ensure that the solver iterations converge asymptotically. In numerical experiments we observe that the sufficient conditions are conservative and in practice the algorithm may tolerate delays when local objectives are not non-strongly convex.

Future work will involve: (i) enabling inter-agent communication within group U or V, via virtual agents (with affine local constraint sets and zero local objectives), as described for example in [1, Sec. 7], with the help of the data exchange server; (ii) investigating acceleration methods for improving the linear convergence rates that are observed in simulations; (iii) tightening the sufficient conditions for algorithm convergence.

Appendix
Convergence Analysis

Lemma 1. *Similar to [18, Lemma 4.1], consider $h_1(x)$ and $h_2(x)$ are convex functions over the convex domain $x \in \mathcal{X}$. We define $\Sigma \succeq 0$ such that $\partial h_1(x_0)^\intercal(x - x_0) + \frac{1}{2}\|x - x_0\|_\Sigma^2 \leq h_1(x) - h_1(x_0), \forall x, x_0 \in \mathcal{X}$. If $\Sigma \succ 0$, this implies $h_1(x)$ is strongly convex. We also define $\hat{x} \overset{def}{=} \arg\min_{x \in \mathcal{X}} h_1(x) + h_2(x)$.*

Therefore we have $\forall x \in \mathcal{X}$:

$$h_1(\hat{x}) - h_1(x) + \frac{1}{2}\|\hat{x} - x\|_\Sigma^2 + \partial h_2(\hat{x})^\intercal(\hat{x} - x) \leq 0. \quad (25)$$

Proof: Since $\hat{x} \overset{\text{def}}{=} \arg\min_{x\in\mathcal{X}} h_1(x) + h_2(x),\ h_1, h_2, \mathcal{X}$ being convex, we have:

$$h_1(\hat{x}) - h_1(x) - \partial h_1(\hat{x})^\intercal(\hat{x} - x) + \frac{1}{2}\|\hat{x} - x\|^2_\Sigma \le 0,$$
$$(\partial h_1(\hat{x}) + \partial h_2(\hat{x}))^\intercal(\hat{x} - x) \le 0.$$

By combining the two equations we obtain (25). ■

Proof of Theorem 2: Since the results 1) and 2) stated in the theorem are equivalent, we explicitly prove only 2).

Part 0, we note that a saddle point of our Lagrangian (7): $\{u_i^*\}, \{v_j^*\}, \{z_{ij}^*, w_{ij}^*, \lambda_{ij}^*\}_{(i,j)\in E}$ has following properties:

$$z_{ij}^* = w_{ij}^* \quad \forall (i,j) \in E \tag{26}$$

$$\begin{aligned}&\mathcal{L}_\Theta(\{u_i^*\}, \{v_j^*\}, \{z_{ij}^*, w_{ij}^*, \lambda_{ij}^*\}_{(i,j)\in E})\\ \le\ &\mathcal{L}_\Theta(\{u_i\}, \{v_j\}, \{z_{ij}, w_{ij}, \lambda_{ij}^*\}_{(i,j)\in E})\\ &\forall \{u_i\}, \{v_j\}, \{z_{ij}\}, \{w_{ij}\} \text{ s.t. } \bigcap \forall\text{(4d)}\forall\text{(4e)}\end{aligned} \tag{27}$$

Part 1, $\forall i \in U$, $\forall k \ge 1$, from (12) and Lemma 1 we have:

$$\begin{aligned}&f_i(u_i^k) - f_i(u_i^*) + \sum_{j\in\mathcal{N}_i}\Big[F_{ij}(z_{ij}^k) - F_{ij}(z_{ij}^*)\\ &+ \Big(\lambda_{ij}^{a_i^2(k)} + \Theta_{ij}(z_{ij}^k - w_{ij}^{a_i^2(k)})\Big)^\intercal(z_{ij}^k - z_{ij}^*)\\ &+ \frac{1}{2}\|z_{ij}^k - z_{ij}^*\|^2_{\Sigma_{ij}^U}\Big] \le 0,\end{aligned} \tag{28}$$

in which with the arbitrarily chosen $\{\lambda_{ij}\}$:

$$\begin{aligned}&\lambda_{ij}^{a_i^2(k)} + \Theta_{ij}(z_{ij}^k - w_{ij}^{a_i^2(k)})\\ =\ &\lambda_{ij}^{a_i^2(k)} + \Theta_{ij}(z_{ij}^k - w_{ij}^k) + \Theta_{ij}(w_{ij}^k - w_{ij}^{a_i^2(k)})\\ \overset{(14)}{=}&\lambda_{ij}^{a_i^2(k)} + \lambda_{ij}^k - \lambda_{ij}^{k-1} + \Theta_{ij}(w_{ij}^k - w_{ij}^{a_i^2(k)})\\ =\ &\lambda_{ij} + (\lambda_{ij}^{a_i^2(k)} - \lambda_{ij}^{k-1}) + (\lambda_{ij}^k - \lambda_{ij})\\ &+ \Theta_{ij}(w_{ij}^k - w_{ij}^{a_i^2(k)}),\end{aligned} \tag{29}$$

and:

$$\begin{aligned}&\Big(\Theta_{ij}(w_{ij}^k - w_{ij}^{a_i^2(k)})\Big)^\intercal(z_{ij}^k - z_{ij}^*)\\ =\ &(w_{ij}^k - w_{ij}^{a_i^2(k)} + w_{ij}^{k-1} - w_{ij}^{k-1})^\intercal\\ &\Theta_{ij}(z_{ij}^k - z_{ij}^* + w_{ij}^k - w_{ij}^k)\\ =\ &(w_{ij}^k - w_{ij}^{k-1})^\intercal\Theta_{ij}(w_{ij}^k - z_{ij}^*)\\ &+ (w_{ij}^k - w_{ij}^{k-1})^\intercal\Theta_{ij}(z_{ij}^k - w_{ij}^k)\\ &+ (w_{ij}^{k-1} - w_{ij}^{a_i^2(k)})^\intercal\Theta_{ij}(z_{ij}^k - z_{ij}^*)\\ \overset{(26)(14)}{=}&(w_{ij}^k - w_{ij}^{k-1})^\intercal\Theta_{ij}(w_{ij}^k - w_{ij}^*)\\ &+ (w_{ij}^k - w_{ij}^{k-1})^\intercal(\lambda_{ij}^k - \lambda_{ij}^{k-1})\\ &+ (w_{ij}^{k-1} - w_{ij}^{a_i^2(k)})^\intercal\Theta_{ij}(z_{ij}^k - z_{ij}^*).\end{aligned} \tag{30}$$

Part 2, similar to Part 1, $\forall j \in V$, $\forall k$, from (13) and Lemma 1 we have:

$$\begin{aligned}&g_j(v_j^k) - g_j(v_j^*) + \sum_{i\in\mathcal{N}_j}\Big[G_{ij}(w_{ij}^k) - G_{ij}(w_{ij}^*)\\ &- \Big(\lambda_{ij}^{b_j^2(k)-1} + \Theta_{ij}(z_{ij}^{b_j^2(k)} - w_{ij}^k)\Big)^\intercal(w_{ij}^k - w_{ij}^*)\\ &+ \frac{1}{2}\|w_{ij}^k - w_{ij}^*\|^2_{\Sigma_{ij}^V}\Big] \le 0,\end{aligned} \tag{31}$$

in which:

$$\begin{aligned}&\lambda_{ij}^{b_j^2(k)-1} + \Theta_{ij}(z_{ij}^{b_j^2(k)} - w_{ij}^k)\\ =\ &\lambda_{ij}^{b_j^2(k)-1} + \Theta_{ij}(z_{ij}^{b_j^2(k)} - w_{ij}^{b_j^2(k)}) + \Theta_{ij}(w_{ij}^{b_j^2(k)} - w_{ij}^k)\\ \overset{(14)}{=}&\lambda_{ij}^{b_j^2(k)} + \Theta_{ij}(w_{ij}^{b_j^2(k)} - w_{ij}^k)\\ =\ &(\lambda_{ij}^{b_j^2(k)} - \lambda_{ij}^k) + (\lambda_{ij}^k - \lambda_{ij}) + \lambda_{ij}\\ &+ \Theta_{ij}(w_{ij}^{b_j^2(k)} - w_{ij}^{k-1}) + \Theta_{ij}(w_{ij}^{k-1} - w_{ij}^k).\end{aligned} \tag{32}$$

Part 3, we combine the equations above as well as (26)(14), sum $\forall i \in U\ \forall j \in V$, and take the average over K steps:

$$\begin{aligned}\frac{1}{K}\sum_{k=1}^K\Bigg[\Delta p^k + \sum_{(i,j)\in E}\Big(&\lambda_{ij}^\intercal(z_{ij}^k - w_{ij}^k)\\ &+ (\lambda_{ij}^k - \lambda_{ij})^\intercal\Theta_{ij}^{-1}(\lambda_{ij}^k - \lambda_{ij}^{k-1}) &\text{(33a)}\\ &+ 2(w_{ij}^k - w_{ij}^{k-1})^\intercal\Theta_{ij}(w_{ij}^k - w_{ij}^*) &\text{(33b)}\\ &+ \frac{1}{2}\|z_{ij}^k - z_{ij}^*\|^2_{\Sigma_{ij}^U} + \frac{1}{2}\|w_{ij}^k - w_{ij}^*\|^2_{\Sigma_{ij}^V}\\ &+ (\lambda_{ij}^{a_i^2(k)} - \lambda_{ij}^{k-1})^\intercal(z_{ij}^k - z_{ij}^*) &\text{(33c)}\\ &+ (w_{ij}^{k-1} - w_{ij}^{a_i^2(k)})^\intercal\Theta_{ij}(z_{ij}^k - z_{ij}^*) &\text{(33d)}\\ &+ (\lambda_{ij}^k - \lambda_{ij}^{b_j^2(k)})^\intercal(w_{ij}^k - w_{ij}^*) &\text{(33e)}\\ &+ (w_{ij}^{k-1} - w_{ij}^{b_j^2(k)})^\intercal\Theta_{ij}(w_{ij}^k - w_{ij}^*) &\text{(33f)}\\ &+ (\lambda_{ij}^k - \lambda_{ij}^{k-1})^\intercal(w_{ij}^k - w_{ij}^{k-1}) &\text{(33g)}\\ \Big)\Bigg] \le 0,\end{aligned}$$

in which:

$$\begin{aligned}\Delta p^k \overset{\text{def}}{=} &\sum_{i\in U}\Big(f_i(u_i^k) - f_i(u_i^*)\Big)\\ &+ \sum_{j\in V}\Big(g_j(v_j^k) - g_j(v_j^*)\Big)\\ &+ \sum_{(i,j)\in E}\Big(F_{ij}(z_{ij}^k) - F_{ij}(z_{ij}^*)\\ &+ G_{ij}(w_{ij}^k) - G_{ij}(w_{ij}^*)\Big).\end{aligned} \tag{34}$$

We note that $\|\lambda_{ij}^a - \lambda_{ij}^b\|^2_{\Theta_{ij}^{-1}} = \|\lambda_{ij}^a\|^2_{\Theta_{ij}^{-1}} + \|\lambda_{ij}^b\|^2_{\Theta_{ij}^{-1}} -$

$2(\lambda_{ij}^a)^\intercal \Theta_{ij}^{-1} \lambda_{ij}^b$, therefore:

$$\begin{aligned}\sum_{k=1}^K (33a) &= \frac{1}{2}\sum_{k=1}^K \Big(\|\lambda_{ij}^k - \lambda_{ij}\|^2_{\Theta_{ij}^{-1}} + \|\lambda_{ij}^k - \lambda_{ij}^{k-1}\|^2_{\Theta_{ij}^{-1}} \\ &\quad - \|\lambda_{ij}^{k-1} - \lambda_{ij}\|^2_{\Theta_{ij}^{-1}} \Big) \\ &= \frac{1}{2}\Big(\|\lambda_{ij}^K - \lambda_{ij}\|^2_{\Theta_{ij}^{-1}} - \|\lambda_{ij}^1 - \lambda_{ij}\|^2_{\Theta_{ij}^{-1}} \\ &\quad + \sum_{k=1}^K \|\lambda_{ij}^k - \lambda_{ij}^{k-1}\|^2_{\Theta_{ij}^{-1}} \Big). \end{aligned} \tag{35}$$

Similarly,

$$\begin{aligned}\sum_{k=1}^K (33b) &= \|w_{ij}^K - w_{ij}^*\|^2_{\Theta_{ij}} - \|w_{ij}^1 - w_{ij}^*\|^2_{\Theta_{ij}} \\ &\quad + \sum_{k=1}^K \|w_{ij}^k - w_{ij}^{k-1}\|^2_{\Theta_{ij}}. \end{aligned} \tag{36}$$

We bound the following term:

$$\begin{aligned}\sum_{k=1}^K (33c) &= \sum_{k=1}^K \sum_{l=a_i^2(k)}^{k-2} (\lambda_{ij}^l - \lambda_{ij}^{l+1})^\intercal (z_{ij}^k - z_{ij}^*) \\ &\le \sum_{k=1}^K \sum_{l=a_i^2(k)}^{k-2} \Big(\frac{1}{2\alpha_{ij}}\|\lambda_{ij}^l - \lambda_{ij}^{l+1}\|^2_{\Theta_{ij}^{-1}} + \frac{\alpha_{ij}}{2}\|z_{ij}^k - z_{ij}^*\|^2_{\Theta_{ij}}\Big) \\ &\le (\tau_i - 1)\sum_{k=1}^K \Big(\frac{1}{\alpha_{ij}}\|\lambda_{ij}^k - \lambda_{ij}^{k-1}\|^2_{\Theta_{ij}^{-1}} + \alpha_{ij}\|z_{ij}^k - z_{ij}^*\|^2_{\Theta_{ij}}\Big), \end{aligned} \tag{37}$$

where $\alpha_{ij} > 0$. To see the 2nd inequality in (37), we count the maximum possible number of duplicates $(\lambda_{ij}^k - \lambda_{ij}^{k+1})\ \forall k$ illustrated in Fig. 8, which is $2\tau_i - 2$.

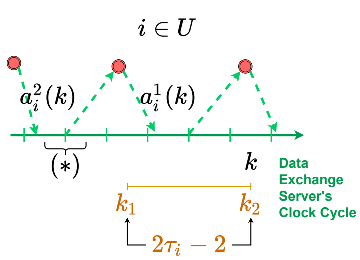

Fig. 8. The $l \to l+1$ couple $(*)$ is duplicated up to $2\tau_i - 2$ times as $k_1 \le k \le k_2$ when deriving the 2nd inequality in (37).

Similarly:

$$\begin{aligned}\sum_{k=1}^K (33d) &\le (\tau_i - 1)\sum_{k=1}^K \Big(\frac{1}{\alpha_{ij}}\|w_{ij}^k - w_{ij}^{k-1}\|^2_{\Theta_{ij}} \\ &\quad + \alpha_{ij}\|z_{ij}^k - z_{ij}^*\|^2_{\Theta_{ij}}\Big), \end{aligned} \tag{38}$$

$$\begin{aligned}\sum_{k=1}^K (33e) &\le \frac{2\tau_j - 1}{2}\sum_{k=1}^K \Big(\frac{1}{\alpha_{ij}}\|\lambda_{ij}^k - \lambda_{ij}^{k-1}\|^2_{\Theta_{ij}^{-1}} \\ &\quad + \alpha_{ij}\|w_{ij}^k - w_{ij}^*\|^2_{\Theta_{ij}}\Big), \end{aligned} \tag{39}$$

$$\begin{aligned}\sum_{k=1}^K (33f) &\le (\tau_j - 1)\sum_{k=1}^K \Big(\frac{1}{\alpha_{ij}}\|w_{ij}^k - w_{ij}^{k-1}\|^2_{\Theta_{ij}} \\ &\quad + \alpha_{ij}\|w_{ij}^k - w_{ij}^*\|^2_{\Theta_{ij}}\Big), \end{aligned} \tag{40}$$

$$\begin{aligned}\sum_{k=1}^K (33g) &\le \frac{1}{2}\sum_{k=1}^K \Big(\frac{\beta_{ij}}{\alpha_{ij}}\|\lambda_{ij}^k - \lambda_{ij}^{k-1}\|^2_{\Theta_{ij}^{-1}} \\ &\quad + \frac{\alpha_{ij}}{\beta_{ij}}\|w_{ij}^k - w_{ij}^{k-1}\|^2_{\Theta_{ij}}\Big). \end{aligned} \tag{41}$$

where $\beta_{ij} > 0$.

Part 4, we re-arrange Part 3 after having substituted (35), (36), (37), (38), (39), (40) and (41) into (33):

$$\begin{aligned}&\frac{1}{K}\sum_{k=1}^K \Bigg[\Delta p^k + \sum_{(i,j)\in E}\Big(\lambda_{ij}^\intercal (z_{ij}^k - w_{ij}^k) \\ &+ \frac{2\tau_i + 2\tau_j - 3 + \beta_{ij} - \alpha_{ij}}{2\alpha_{ij}}\|\lambda_{ij}^k - \lambda_{ij}^{k-1}\|^2_{\Theta_{ij}^{-1}} &\text{(42a)} \\ &+ \frac{2\tau_i + 2\tau_j - 4 + \alpha_{ij}^2/\beta_{ij} - 2\alpha_{ij}}{2\alpha_{ij}}\|w_{ij}^k - w_{ij}^{k-1}\|^2_{\Theta_{ij}} &\text{(42b)} \\ &+ \frac{1}{2}(z_{ij}^k - z_{ij}^*)^\intercal\big[(4\tau_i - 4)\alpha_{ij}\Theta_{ij} - \Sigma_{ij}^U\big](z_{ij}^k - z_{ij}^*) &\text{(42c)} \\ &+ \frac{1}{2}(w_{ij}^k - w_{ij}^*)^\intercal\big[(4\tau_j - 3)\alpha_{ij}\Theta_{ij} - \Sigma_{ij}^V\big](w_{ij}^k - w_{ij}^*) \\ & &\text{(42d)} \\ &\Big)\Bigg] + \frac{1}{2K}\sum_{(i,j)\in E}\Big(\|\lambda_{ij}^K - \lambda_{ij}\|^2_{\Theta_{ij}^{-1}} - \|\lambda_{ij}^1 - \lambda_{ij}\|^2_{\Theta_{ij}^{-1}} \\ &+ \|w_{ij}^K - w_{ij}^*\|^2_{\Theta_{ij}} - \|w_{ij}^1 - w_{ij}^*\|^2_{\Theta_{ij}}\Big) \le 0. \end{aligned}$$

We choose $\beta_{ij} = \alpha_{ij} - (2\tau_i + 2\tau_j - 3) \ge 0$ to make (42a) $= 0$. Solve (42b) ≤ 0 and $\alpha_{ij} - (2\tau_i + 2\tau_j - 3) \ge 0$ for α_{ij}:

$$\alpha_{ij} \ge 1 + \frac{1}{2}\big(3\tau_{ij} + \sqrt{5\tau_{ij}^2 + 8\tau_{ij} + 4}\big), \tag{43}$$

where $\tau_{ij} \overset{\text{def}}{=} 2\tau_i + 2\tau_j - 4$.

Let $\alpha_{ij} = 1 + \frac{1}{2}(3\tau_{ij} + \sqrt{5\tau_{ij}^2 + 8\tau_{ij} + 4})$. Solve (42c) ≤ 0, (42d) ≤ 0 for Θ_{ij}, and we obtain the conditions (a)-(e) in Assumption 4.

Part 5, with Assumption 4 being imposed, our (42) is therefore reduced to:

$$\begin{aligned}&\frac{1}{K}\sum_{k=1}^K \Big[\Delta p^k + \sum_{(i,j)\in E}\lambda_{ij}^\intercal (z_{ij}^k - w_{ij}^k)\Big] \\ &\quad - \frac{1}{2K}\sum_{(i,j)\in E}\Big(\|\lambda_{ij}^1 - \lambda_{ij}\|^2_{\Theta_{ij}^{-1}} + \|w_{ij}^1 - w_{ij}^*\|^2_{\Theta_{ij}}\Big) \\ &= \frac{1}{K}\sum_{k=1}^K \Delta p^k + \sum_{(i,j)\in E}\Big(\lambda_{ij}^\intercal (\bar{z}_{ij}^K - \bar{w}_{ij}^K) \\ &\quad - \frac{1}{2K}\big(\|\lambda_{ij}^1 - \lambda_{ij}\|^2_{\Theta_{ij}^{-1}} + \|w_{ij}^1 - w_{ij}^*\|^2_{\Theta_{ij}}\big)\Big) \\ &\le 0, \end{aligned} \tag{44}$$

in which $\bar{x}^K$ denotes the running average: $\sum_k x^k$. Since we have convex cost functions,

$$\begin{aligned}\frac{1}{K}\sum_{k=1}^{K}\Delta p^k \geq &\sum_{i\in U}\Big(f_i(\bar{u}_i^K)-f_i(u_i^*)\Big)\\ &+\sum_{j\in V}\Big(g_j(\bar{v}_j^K)-g_j(v_j^*)\Big)\\ &+\sum_{(i,j)\in E}\Big(F_{ij}(\bar{z}_{ij}^K)-F_{ij}(z_{ij}^*)\\ &+G_{ij}(\bar{w}_{ij}^K)-G_{ij}(w_{ij}^*)\Big) \stackrel{\text{def}}{=} \Delta\bar{p}^K.\end{aligned} \tag{45}$$

From (27), we have:

$$\Delta\bar{p}^K+\sum_{(i,j)\in E}{\lambda_{ij}^*}^\intercal(\bar{z}_{ij}^K-\bar{w}_{ij}^K)\geq 0. \tag{46}$$

We combine (44) and (45). From the result we subtract (46) to get:

$$\begin{aligned}&\sum_{(i,j)\in E}\Big((\lambda_{ij}-\lambda_{ij}^*)^\intercal(\bar{z}_{ij}^K-\bar{w}_{ij}^K)\\ &-\frac{1}{2K}\big(\|\lambda_{ij}^1-\lambda_{ij}\|^2_{\Theta_{ij}^{-1}}+\|w_{ij}^1-w_{ij}^*\|^2_{\Theta_{ij}}\big)\Big)\leq 0.\end{aligned} \tag{47}$$

Since $\{\lambda_{ij}\}$ are arbitrarily chosen, we choose $\lambda_{ij}=\lambda_{ij}^*+e_{ij}$, $e_{ij}=\frac{\bar{z}_{ij}^K-\bar{w}_{ij}^K}{\|\bar{z}_{ij}^K-\bar{w}_{ij}^K\|_2}$ and substitute into (47) to obtain:

$$\begin{aligned}&\sum_{(i,j)\in E}\|\bar{z}_{ij}^K-\bar{w}_{ij}^K\|_2\\ \leq&\frac{1}{K}\sum_{(i,j)\in E}\Big(\frac{1}{2}\max_{\|e_{ij}\|_2=1}\|\lambda_{ij}^1-\lambda_{ij}^*-e_{ij}\|^2_{\Theta_{ij}^{-1}}\\ &+\|w_{ij}^1-w_{ij}^*\|^2_{\Theta_{ij}}\Big)\\ \stackrel{\text{def}}{=}&\frac{C_1}{K}.\end{aligned} \tag{48}$$

We also have:

$$\begin{aligned}&\Delta\bar{p}^K+\sum_{(i,j)\in E}{\lambda_{ij}^*}^\intercal(\bar{z}_{ij}^K-\bar{w}_{ij}^K)\\ \stackrel{(46)}{=}&|\Delta\bar{p}^K+\sum_{(i,j)\in E}{\lambda_{ij}^*}^\intercal(\bar{z}_{ij}^K-\bar{w}_{ij}^K)|\\ \geq&|\Delta\bar{p}^K|-\sum_{(i,j)\in E}|{\lambda_{ij}^*}^\intercal(\bar{z}_{ij}^K-\bar{w}_{ij}^K)|\\ \geq&|\Delta\bar{p}^K|-\sum_{(i,j)\in E}\|\lambda_{ij}^*\|_\infty\|\bar{z}_{ij}^K-\bar{w}_{ij}^K\|_1\\ \stackrel{(*)}{\geq}&|\Delta\bar{p}^K|-\sum_{(i,j)\in E}\|\lambda_{ij}^*\|_\infty\sqrt{\dim(\lambda_{ij})}\|\bar{z}_{ij}^K-\bar{w}_{ij}^K\|_2\\ \stackrel{(48)}{\geq}&|\Delta\bar{p}^K|-\frac{C_1C_2}{K},\end{aligned} \tag{49}$$

in which $(*)$ is due to norm equivalence.

Finally from (49),

$$\begin{aligned}|\Delta\bar{p}^K|&\leq\frac{C_1C_2}{K}+\Delta\bar{p}^K+\sum_{(i,j)\in E}{\lambda_{ij}^*}^\intercal(\bar{z}_{ij}^K-\bar{w}_{ij}^K)\\ &\stackrel{(44)(45)}{\leq}\frac{1}{2K}\big(C_3+\|\lambda_{ij}^1-\lambda_{ij}^*\|^2_{\Theta_{ij}^{-1}}+\|w_{ij}^1-w_{ij}^*\|^2_{\Theta_{ij}}\big)\\ &=\frac{C}{K}.\end{aligned} \tag{50}$$

We output the following results: $\{\bar{u}_i^K\},\{\bar{v}_j^K\},\{\bar{z}_{ij}^K,\bar{w}_{ij}^K\}$.

Feasibility check: Note that (48) shows as $K\to\infty$ the output satisfies dualised constraints (4c). Since taking the running average is a convex combination, it also satisfies all the the local constraints (4d)(4e).

Optimality: (50) shows as $K\to\infty$ the output minimises the cost function of $\mathcal{P}$, and hence converges to a minimiser of our problem. ■

References

[1] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," 2010.

[2] A. Nedić, "Convergence rate of distributed averaging dynamics and optimization in networks," *Foundations and Trends in Systems and Control*, vol. 2, no. 1, pp. 1–100, 2015.

[3] T. Yang, X. Yi, J. Wu, Y. Yuan, D. Wu, Z. Meng, Y. Hong, H. Wang, Z. Lin, and K. H. Johansson, "A survey of distributed optimization," *Annual Reviews in Control*, vol. 47, pp. 278–305, 2019.

[4] A. Nedić and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[5] A. Nedic, A. Ozdaglar, and P. A. Parrilo, "Constrained consensus and optimization in multi-agent networks," *IEEE Transactions on Automatic Control*, vol. 55, no. 4, pp. 922–938, 2010.

[6] P. Lin, W. Ren, and Y. Song, "Distributed multi-agent optimization subject to nonidentical constraints and communication delays," *Automatica*, vol. 65, pp. 120–131, 3 2016.

[7] A. Nedić and A. Olshevsky, "Distributed optimization over time-varying directed graphs," *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 601–615, 2015.

[8] K. J. Arrow, L. Hurwicz, and H. Uzawa, *Studies In Linear And Non-Linear Programming*. Stanford mathematical studies in the social sciences, Stanford University Press, 1972.

[9] R. Glowinski and A. Marroco, "Sur l'approximation, par Éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires," *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique*, vol. 9, no. R2, pp. 41–76, 1975.

[10] B. He and X. Yuan, "On the o(1/n) convergence rate of the douglas–rachford alternating direction method," *SIAM Journal on Numerical Analysis*, vol. 50, pp. 700–709, 1 2012.

[11] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the ADMM in decentralized consensus optimization," *IEEE Transactions on Signal Processing*, vol. 62, 4 2014.

[12] W. Deng and W. Yin, "On the global and linear convergence of the generalized alternating direction method of multipliers," *Journal of Scientific Computing*, vol. 66, no. 3, pp. 889–916, 2016.

[13] D. Davis and W. Yin, "Convergence rate analysis of several splitting schemes," in *Splitting Methods in Communication, Imaging, Science, and Engineering*, pp. 115–163, Springer, 6 2016.

[14] Q. Ho, J. Cipar, H. Cui, J. K. Kim, S. Lee, P. B. Gibbons, G. A. Gibson, G. R. Ganger, and E. P. Xing, "More effective distributed ml via a stale synchronous parallel parameter server," *Advances in Neural Information Processing Systems*, no. 1, pp. 1–9, 2013.

[15] Z. Peng, Y. Xu, M. Yan, and W. Yin, "Arock: An algorithmic framework for asynchronous parallel coordinate updates," *SIAM Journal on Scientific Computing*, vol. 38, pp. A2851–A2879, 1 2016.

[16] T. H. Chang, M. Hong, W. C. Liao, and X. Wang, "Asynchronous distributed ADMM for large-scale optimization - part i: Algorithm and convergence analysis," *IEEE Transactions on Signal Processing*, vol. 64, pp. 3118–3130, 6 2016.

[17] J. N. Tsitsiklis, D. P. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Transactions on Automatic Control*, vol. 31, no. 9, pp. 803–812, 1986.

[18] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel And Distributed Computation: Numerical Methods*. Athena Scientific, 1997.

[19] A. Nedić, D. P. Bertsekas, and V. S. Borkar, "Distributed asynchronous incremental subgradient methods," *Studies in Computational Mathematics*, vol. 8, no. C, pp. 381–407, 2001.

[20] A. Nedić and A. Ozdaglar, "Convergence rate for consensus with delays," *Journal of Global Optimization*, vol. 47, no. 3, pp. 437–456, 2010.

[21] A. Agarwal and J. C. Duchi, "Distributed delayed stochastic optimization," *Proceedings of the IEEE Conference on Decision and Control*, pp. 5451–5452, 2012.

[22] I. Notarnicola and G. Notarstefano, "Asynchronous distributed optimization via randomized dual proximal gradient," *IEEE Transactions on Automatic Control*, vol. 62, no. 5, pp. 2095–2106, 2017.

[23] R. Hannah and W. Yin, "On unbounded delays in asynchronous parallel fixed-point algorithms," *Journal of Scientific Computing*, vol. 76, no. 1, pp. 299–326, 2018.

[24] X. Lian, W. Zhang, C. Zhang, and J. Liu, "Asynchronous decentralized parallel stochastic gradient descent," *35th International Conference on Machine Learning, ICML 2018*, vol. 7, pp. 4745–4767, 2018.

[25] K. Mishchenko, F. Iutzeler, J. Malick, and M.-R. R. Amini, "A delay-tolerant proximal-gradient algorithm for distributed learning," *35th International Conference on Machine Learning, ICML 2018*, vol. 8, pp. 5774–5788, 2018.

[26] E. Wei and A. Ozdaglar, "On the o(1/k) convergence of asynchronous distributed alternating direction method of multipliers," *2013 IEEE Global Conference on Signal and Information Processing, GlobalSIP 2013 - Proceedings*, pp. 551–554, 2013.

[27] F. Iutzeler, P. Bianchi, P. Ciblat, and W. Hachem, "Asynchronous distributed optimization using a randomized alternating direction method of multipliers," *Proceedings of the IEEE Conference on Decision and Control*, pp. 3671–3676, 2013.

[28] R. Zhang and J. T. Kwok, "Asynchronous distributed ADMM for consensus optimization," *31st International Conference on Machine Learning, ICML 2014*, vol. 5, no. 2, pp. 3689–3697, 2014.

[29] L. Fang and Y. Lei, "An asynchronous distributed ADMM algorithm and efficient communication model," *Proceedings - 2016 IEEE 14th International Conference on Dependable, Autonomic and Secure Computing, DASC 2016, 2016 IEEE 14th International Conference on Pervasive Intelligence and Computing, PICom 2016, 2016 IEEE 2nd International Conference on Big Data*, no. 2, pp. 136–140, 2016.

[30] S. Kumar, R. Jain, and K. Rajawat, "Asynchronous optimization over heterogeneous networks via consensus ADMM," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 3, no. 1, pp. 114–129, 2017.

[31] J. Zhou and Y. Lei, "Asynchronous group-based ADMM algorithm under efficient communication structure," in *2018 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, pp. 135–140, IEEE, 12 2018.

[32] M. Ma, J. Ren, G. B. Giannakis, and J. Haupt, "Fast asynchronous decentralized optimization: Allowing multiple masters," in *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 633–637, IEEE, 11 2018.

[33] S. Jiang, Y. Lei, S. Wang, and D. Wang, "An asynchronous ADMM algorithm for distributed optimization with dynamic scheduling strategy," in *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pp. 1–8, IEEE, 8 2019.

[34] T. H. Chang, M. Hong, W. C. Liao, and X. Wang, "Asynchronous distributed alternating direction method of multipliers: Algorithm and convergence analysis," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2016-May, pp. 4781–4785, 2016.

[35] T. H. Chang, M. Hong, W. C. Liao, and X. Wang, "Asynchronous distributed ADMM for large-scale optimization- part i: Algorithm and convergence analysis - online," *arXiv*, p. 38, 9 2015.

[36] T. H. Chang, W. C. Liao, M. Hong, and X. Wang, "Asynchronous distributed ADMM for large-scale optimization - part ii: Linear convergence analysis and numerical performance," *IEEE Transactions on Signal Processing*, vol. 64, pp. 3131–3144, 6 2016.

[37] M. Hong, "A distributed, asynchronous, and incremental algorithm for nonconvex optimization: An ADMM approach," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 3, pp. 935–945, 2018.

[38] A. Falsone, I. Notarnicola, G. Notarstefano, and M. Prandini, "Tracking-ADMM for distributed constraint-coupled optimization," *Automatica*, vol. 117, p. 108962, 2020.

[39] V. Khatana and M. V. Salapaka, "D-distADMM: A o(1/k) distributed ADMM for distributed optimization in directed graph topologies," *Proceedings of the IEEE Conference on Decision and Control*, vol. 2020-Decem, no. Cdc, pp. 2992–2997, 2020.

[40] W. Jiang and T. Charalambous, "Distributed alternating direction method of multipliers using finite-time exact ratio consensus in digraphs," *2021 European Control Conference, ECC 2021*, pp. 2205–2212, 2021.

[41] N. Bastianello, R. Carli, L. Schenato, and M. Todescato, "Asynchronous distributed optimization over lossy networks via relaxed ADMM: Stability and linear convergence," *IEEE Transactions on Automatic Control*, vol. 66, no. 6, pp. 2620–2635, 2021.

[42] A. Mohammadi and A. Kargarian, "Learning-aided asynchronous ADMM for optimal power flow," *IEEE Transactions on Power Systems*, vol. 37, no. 3, pp. 1671–1681, 2022.

[43] M. Mesbahi and M. Egerstedt, *Graph Theoretic Methods in Multiagent Networks*. Princeton University Press, 12 2010.

[44] J. M. Morales, A. J. Conejo, and J. Pérez-Ruiz, "Economic valuation of reserves in power systems with high penetration of wind power," *IEEE Transactions on Power Systems*, vol. 24, no. 2, pp. 900–910, 2009.

[45] F. Rey, Z. Pan, A. Hauswirth, and J. Lygeros, "Fully decentralized ADMM for coordination and collision avoidance," *2018 European Control Conference, ECC 2018*, pp. 825–830, 2018.

[46] ISO/IEC 7498-1:1994, "Information technology – Open Systems Interconnection – Basic Reference Model," tech. rep., International Organization for Standardization, Geneva, CH, 1996.

ZhouDan Pan obtained a Bachelor's degree in Automotive Engineering from Politecnico di Torino in 2011, and an MSc in Mechatronics, Robotics, and Automation Engineering from ETH Zürich in 2017. He is currently studying for a DPhil degree with the Control Group, Department of Engineering Science, University of Oxford, and is a member of St Peter's College, Oxford. His research is concerned with distributed control and optimisation, robust optimisation, and power systems applications.

Mark Cannon obtained the MEng and DPhil degrees from the University of Oxford in 1993 and 1998, and the SM degree from Massachusetts Institute of Technology, Cambridge, MA, in 1995. His current position is Associate Professor of Engineering Science at the University of Oxford and he is a Fellow of St John's College, Oxford. His research interests are in robust, stochastic and adaptive model predictive control, robust and distributed optimization, and applications in optimal energy management, trajectory optimization under uncertainty, electroceuticals for neurological disorders, and bioprocess control.