On Distributed Solution of Ill-Conditioned System of Linear Equations under Communication Delays

Kushal Chakrabarti[1], Nirupam Gupta[2] and Nikhil Chopra[3]

***Abstract*— This paper considers a distributed solution for a system of linear equations. The underlying peer-to-peer communication network is assumed to be undirected, however, the communication links are subject to potentially large but constant delays. We propose an algorithm that solves a distributed least-squares problem, which is equivalent to solving the system of linear equations. Effectively, the proposed algorithm is a pre-conditioned version of the traditional consensus-based distributed gradient descent (DGD) algorithm. We show that the accuracy of the solution obtained by the proposed algorithm is better than the DGD algorithm, especially when the system of linear equations is ill-conditioned.**

I. INTRODUCTION

Two of the major existing challenges in solving a set of linear equations, $Ax = b$, are *high dimensionality* and *ill-conditioning*. When the dimension of b (or A) is large, the workload of solving this problem can be distributed by splitting among a number of agents [1], [2], [3], [4]. The focus of existing distributed algorithms is primarily on accuracy and efficiency in terms of required computational capacity and memory.

In this paper we solve the system of equations

$$Ax = b, \tag{1}$$

where rows of $A \in \mathbb{R}^{N \times n}$ and corresponding elements of $b \in \mathbb{R}^N$ are distributed among m agents. Each agent knows only a subset (A^i, b^i) of the rows in (A, b) so that $A = \begin{bmatrix}(A^1)^T & \dots & (A^m)^T\end{bmatrix}^T \in \mathbb{R}^{N \times n}$ and $b = \begin{bmatrix}(b^1)^T & \dots & (b^m)^T\end{bmatrix}^T \in \mathbb{R}^N$, where $A^i \in \mathbb{R}^{n^i \times n}$, $b^i \in \mathbb{R}^{n^i}$ and $N = \sum_{i=1}^m n^i$. The agents can communicate with each other over a peer-to-peer network.

The network is represented as an undirected fixed graph $\mathbb{G} = (\mathbb{V}, \mathbb{E})$, with m nodes $\mathbb{V} = \{1, \dots, m\} =: [m]$ representing the agents and the set of edges $\mathbb{E}$, where an edge $(i, j) \in \mathbb{E}$ between two nodes $i, j \in \mathbb{V}$ represents that the agent i and agent j are immediate neighbours. It is assumed that there is no self edge from a node $i \in \mathbb{V}$ to itself.

We consider two major challenges:

- ill-conditioned matrix A,
- delay in the communication links.

*This work is being carried out as a part of the Pipeline System Integrity Management Project, which is supported by the Petroleum Institute, Khalifa University of Science and Technology, Abu Dhabi, UAE.

[1]Department of Electrical and Computer Engineering, University of Maryland, College Park, MD 20742, USA (kchakrabarti0@gmail.com).

[2]Department of Computer Science, Georgetown University, Washington, D.C. 20057, USA (nirupam.gupta@georgetown.edu).

[3]Department of Mechanical Engineering, University of Maryland, College Park, MD 20742, USA (nchopra@umd.edu).

The communication delay from an agent i to agent j is modeled as τ^{ij}, where $\tau^{ij} = \tau^{ji} > 0$ is constant for any edge $(i, j) \in \mathbb{E}$. Other approaches to study the delay robustness problem can be found in [3], [5], [6].

The considered problem has also been addressed in prior works [2], [3], [5], [7], [8]. Initially [2] and later on [3] consider directed time-varying networks. [3] proves the global convergence of a projection-based asynchronous algorithm, with the assumption of extended neighbour graphs being repeatedly jointly strongly connected and bounded delays, and provides an upper bound on the convergence rate. Random communications have been considered in [5], [8] and algorithms have been provided with almost sure convergence to a solution of (1). [7] proves convergence of a communication-efficient extension of the algorithm in [2]. In the algorithm proposed in [7], each agent also needs to share the number of their neighbours. We guarantee convergence in case of a deterministic network topology, without assuming a bound on the constant delays and by the agents sharing only their estimates with neighbors.

The solution of (1) can also be obtained by solving a least-squares problem, such as in [3], [9], [10], [11]. When (1) is not solvable, the least-squares formulation has an advantage of finding the solution that best fits (1). The least-squares problem can also be solved by general distributed optimization algorithms that have been discussed in the literature. However, ill-conditioning of A poses an additional challenge when there are communication delays between the agents. Existing distributed optimization algorithms [6], [12], [13], [14], [15] fare poorly if A is ill-conditioned. The algorithms in [12], [13], [14] require decreasing stepsize, which leads to slower convergence. [6] needs additional variables to be shared with neighbouring agents. [14] requires prior information on the delays. The algorithm proposed in [15] globally converges under strict convexity of each agent's cost function. Moreover, when A is ill-conditioned, these gradient-based optimization approaches suffer from poor convergence rate or may even converge to an undesired point. We follow the same distributed optimization approach, however, our algorithm converges faster without requiring strict convexity of the local costs and shares only the estimates between neighbours. The key ingredient of our proposed approach is local pre-conditioning, which is obtained by solving an appropriate Lyapunov equation. Additionally, the proposed algorithm does not require any information about the heterogeneous communication delays, albeit that they are constant.

TABLE I: Notations

Symbol	Meaning
$\mathcal{N}^i$	neighbor set of node i
$\eta(A)$	nullity of matrix A
$\dot{x}$	first derivative of x w.r.t time t
$\|S\|$	cardinality of a set S
$A \succ 0$	matrix A is positive definite
I_n	identity matrix of order n
$\mathcal{L}_\infty$	set of bounded functions
S_+^n	set of symmetric positive semi-definite matrices of order n
S_{++}^n	set of symmetric positive definite matrices of order n

A. Summary of Contributions:

We solve (1) by solving the least-squares problem,

$$\underset{x^i \in \mathbb{R}^n, \forall i \in [m]}{minimize} \sum_{i=1}^{m} \frac{1}{2} \left\| A^i x^i - b^i \right\|^2$$
$$\text{subject to: } x^i = x^j,\ \forall j \in \mathcal{N}^i,\ \forall i \in \mathbb{V}. \tag{2}$$

[16] uses a similar approach with stricter restrictions on A^i.

For now, we make the following assumptions:

Assumption 1: There is no noise in the measured data.

Assumption 2: $\mathbb{G}$ is a connected graph.

Assumption 3: $\eta(A) = 0$, whereas for each agent $i \in \mathbb{V}$, $\eta(A^i) > 0$.

***Lemma** 1:* Under Assumption 2, (2) and the following optimization problems are equivalent:

$$\underset{x \in \mathbb{R}^n}{minimize} \frac{1}{2} \left\| Ax - b \right\|^2. \tag{3}$$

Proof: See Appendix A. ∎

Problem (2) can be solved using existing distributed optimization algorithms, such as [17], [18], [19]. This fact combined with Lemma 1 and equivalency of problem (1) and (3) enable us to formulate the following distributed solution of (1). Each agent $i \in \mathbb{V}$ knows only its own cost function based on (A^i,b^i) and minimizes that cost cooperatively, while sharing only its solution with its neighbour agents, as represented in problem (2).

***Remarks** 1:* The aggregate cost $(1/2) \left\| Ax - b \right\|^2$ is convex. If $\eta(A) = 0$, then A has full column rank. So the aggregate cost is strictly convex and, hence, has a unique minima. Thus, Assumption 3 implies that (1) has a unique solution x^* satisfying $Ax^* = b$.

Compared to the existing works that address communication delays in distributed optimization, the **major contributions** of our proposed algorithm are follows:

1) Higher robustness to ill-conditioning of A, unlike [12], [13], [14], [15].
2) Local agents' costs need not be strictly convex, i.e. A^i need not be full-rank, unlike [12], [15].
3) The communication delays are apriori unknown, unlike [14].
4) Addressing communication delays in solving least-squares formulation, unlike [2], [3].

II. SHORTCOMING OF DISTRIBUTED GRADIENT METHOD

Our goal is to design a distributed optimization algorithm that solves (2) for ill-conditioned matrix A and in the presence of constant communication delays in the network. In this section we show that the distributed gradient descent method, which is the base of gradient-based distributed optimization methods, can not solve the distributed optimization problem (2) in the presence of communication delay.

Distributed gradient descent (DGD) [19] is a consensus-based iterative algorithm for solving distributed optimization problems. Here each agent i maintains a local estimate of global minima, let it be denoted by $x^i(t)$ for iteration t. All the agents update their local estimates using their local costs' gradients and *current* local estimates of their neighbors. Theoretically, under Assumption 2, if there is no communication delay in the network then the agents' estimates converge to common global minima. However, in the presence of communication delays an agent receives past (and not the current) local estimates from its neighbors. Consequentially, in DGD the agents' local estimates need not converge to global minima or even reach consensus, as shown below. Specifically, considering delay in inter-agent communications, each local agent i updates its estimate for x^* according to

$$\begin{aligned} \dot{x}^i(t) = &- \delta(t)(A^i)^T (A^i x^i(t) - b^i) \\ &+ \sum_{j=1}^{m} w^{ij} (x^j(t - \tau^{ji}) - x^i(t)), \end{aligned} \tag{4}$$

where $x^i(t) \in \mathbb{R}^n$ is estimate computed by agent i after t iterations, $\delta(t) > 0$ is a stepsize along descent direction after t iterations, $w^{ij} \in \mathbb{R}$ is the weight assigned by agent i on the estimate computed by agent j and $\tau^{ii} = 0$. The weights w^{ij} satisfy Assumption 1 of [19].

Example 1. Consider the following numerical example: $A^1 = \begin{bmatrix} 1 & 0.99 \end{bmatrix}$, $A^2 = \begin{bmatrix} 1 & 1.01 \end{bmatrix}$, $b^1 = b^2 = 1$, $\tau^{ij} = 2$, $n = m = 2$ and the network graph is a cycle.

We implement algorithm (4) with parameters $\delta(t) = 1/t$; $w^{ij} = 1/2$ if $j \in \mathcal{N}^i \cup \{i\}$ and $w^{ij} = 0$ otherwise. It can be seen from Fig. 1 that, DGD fails to estimate x^* for the above problem where A has condition number 200. Also, convergence depends on the initial values of the agent states (inset of Fig. 1).

III. PROPOSED ALGORITHM

In this section, we propose a distributed optimization algorithm for solving (1) that is robust to communication delays even when the matrix A is ill-conditioned.

The proposed algorithm is a modification of proportional-integral (PI) consensus-based gradient descent method [15], wherein we (a) remove the integral terms from the algorithm and (b) introduce pre-conditioning at every node. It is evident from the analysis of the proposed method, that integral terms are not required for linear problems. The pre-conditioning matrices have been introduced in the algorithm to take care of ill-conditioning of A. Even in the synchronous

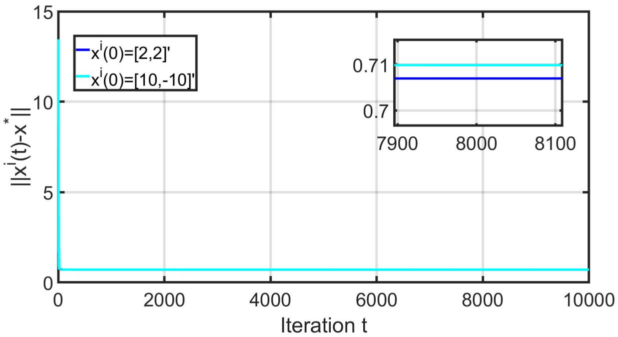

Fig. 1: Temporal evolution of error norm for estimate of each agent $\left\|x^i(t) - x^*\right\|$ for Example 1, under DGD.

setting without any communication delay, ill-conditioning of A slows down convergence or, even worse, can make the agents converge to an undesired solution. Proper choice of conditioning matrices can tolerate ill-conditioning.

For each agent $i \in [m]$, we denote the local gradients as

$$\phi^i(x^i) := (A^i)^T(A^i x^i - b^i). \tag{5}$$

Define local preconditioner matrix K^i which is the solution of the Lyapunov equation [20]

$$-N^i K^i - K^i N^i = -2kI_n, \tag{6}$$

with $k > 0$ and $N^i = (A^i)^T A^i + |\mathcal{N}^i| I_n$. Since $N^i \succ 0$ and $k > 0$, (6) has a unique symmetrical solution that is specified later in the convergence analysis of the algorithm. Using the above definitions, we describe Algorithm 1 below.

Algorithm 1

1: **for** $t = 0, 1, 2, \ldots$ **do**
2: **for** each agent $i \in [m]$ **do**
3: receive $x^j(t - \tau^{ji})$, $\forall j \in \mathcal{N}^i$
4: update estimate

$$\dot{x}^i(t) = K^i[\eta(\eta I_n + K^i)^{-1} \sum_{j \in \mathcal{N}^i} (x^j(t - \tau^{ji}) - x^i(t)) - \phi^i(x^i(t))]. \tag{7}$$

5: transmit updated estimate to all $j \in \mathcal{N}^i$
6: **end for**
7: **end for**

In step 3 of Algorithm 1, each $x^j(t - \tau^{ji})$ can be set to any value for $t < \tau^{ji}$. The choice of preconditioner matrices K^i addresses ill-conditioning of the matrix A, helping the local estimates to converge quickly to the desired solution.

A. Convergence Analysis

***Lemma** 2:* For every $i \in [m]$, $\bar{K}^i := (K^i + \eta I_n)^{-1}(K^i - \eta I_n)$ is Schur for any $\eta > 0$.

Proof: See Appendix B. ■

In order to establish convergence of Algorithm 1, we develop a framework based on [15] and pre-conditioning. The analysis of our algorithm then easily follows from this framework. Note that, the following framework is solely for analysis purposes.

Define consensus terms for each agent:

$$v^i = \sum_{j \in \mathcal{N}^i} v^{ij}, \; v^{ij} = K^i(r_{ij} - x^i). \tag{8}$$

In this framework, r_{ij} is an external input to agent i from its neighbor $j \in \mathcal{N}^i$. In order to evaluate these r_{ij}'s, we define the following transformations [15]:

$$\vec{s}_{ij} = \frac{1}{\sqrt{2\eta}}(-v^{ij} + \eta r_{ij}), \overleftarrow{s}_{ij} = \frac{1}{\sqrt{2\eta}}(v^{ij} + \eta r_{ij}), \tag{9}$$

$$\overleftarrow{s}_{ji} = \frac{1}{\sqrt{2\eta}}(v^{ji} + \eta r_{ji}), \; \vec{s}_{ji} = \frac{1}{\sqrt{2\eta}}(-v^{ji} + \eta r_{ji}), \tag{10}$$

where $\eta > 0$. Information about agent j's estimate is contained in v^{ji} which is sent to neighbor agent $i \in \mathcal{N}^j$ in the form of $\vec{s}_{ji}$. Considering the delay model defined in Section 1, these communication variables are related as

$$\overleftarrow{s}_{ji}(t) = \vec{s}_{ij}(t - \tau^{ij}), \overleftarrow{s}_{ij}(t) = \vec{s}_{ji}(t - \tau^{ji}). \tag{11}$$

Upon receiving $\overleftarrow{s}_{ij}$, which is the variable $\vec{s}_{ji}$ sent by agent j but delayed in time by τ^{ji}, agent i then recovers r_{ij} from it using (9). Hence, r_{ij} contains information about delayed estimates of its neighbor j. These transformations are commonly known as scattering transformation [15]. We assume, $\vec{s}_{ij}(t) = \vec{s}_{ji}(t) = 0 \; \forall t < 0$.

Using the above definitions, we propose the following lemma which will be useful in proving convergence of Algorithm 1.

***Lemma** 3:* Consider the following dynamics for each agent $i \in [m]$:

$$\dot{x}^i = v^i - K^i \phi^i(x^i), \tag{12}$$

the transformation (9)-(10) and delays (11) for $j \in \mathcal{N}^i$, $\forall i$. If Assumptions 1-3 hold, then $x^i \to x^* \; \forall i \in [m]$ as $t \to \infty$.

Proof: See Appendix C. ■

The following theorem shows global convergence of Algorithm 1.

***Theorem** 1:* Consider Algorithm 1. If Assumptions 1-3 hold, then $x^i \to x^* \; \forall i \in [m]$ as $t \to \infty$.

Proof: See Appendix D. ■

IV. SIMULATION RESULTS

Example 2. Consider the following numerical example. The matrix A is a real symmetric positive definite matrix from the benchmark dataset[1] *"bcsstm19"* which is part of a suspension bridge problem. The dimension of A is $N = n = 817$ and its condition number is $2.337333 * 10^5$. The rows of A and the corresponding rows of b are distributed among $m = 5$ agents, so that four of them know 163 such rows and the remaining 165 rows are known by the fifth agent. The network is assumed to be of ring topology and $\tau^{ij} = 5$. We set $b = Ax^*$ where $x^* = \begin{bmatrix} 1 & 1 & \ldots & 1 \end{bmatrix}^T$ is the unique solution.

[1] https://sparse.tamu.edu

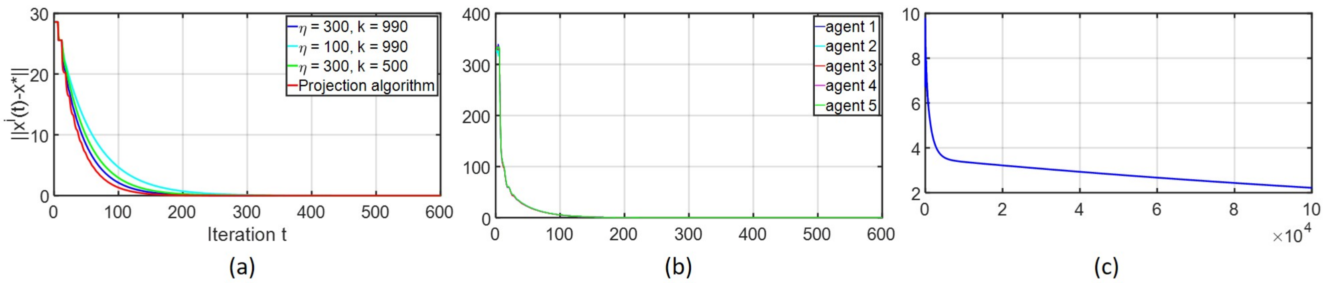

Fig. 2: Temporal evolution of error norm for estimate of each agent $\|x^i(t) - x^*\|$ for Example 2, under (a) Algorithm 1 and projection algorithm [3], (b) Algorithm 1, (c) under the algorithm in [15]. (a) $x^i(0) = [0, \ldots, 0]^T \ \forall i$ in each of the cases. (b) $\eta = 300, k = 990$, $x^i(0)$ are randomly initialized within $[-20, 20]$. (c) $x^i(0) = [3, \ldots, 3]^T \ \forall i$.

We implement Algorithm 1 on this numerical example. The differential equations governing x^i's are solved numerically with small stepsizes $h = 10^{-3}$. Each agent converge to the desired solution x^* irrespective of their initial choice of estimate (ref. Fig. 2) and convergence rate can be changed by tuning the algorithm parameters (ref. Fig. 2a). We compare this result with that of two other algorithms in [15] and [3]. The algorithm in [15] has slower convergence rate (ref. Fig. 2c) due of ill-conditioning of the matrix A, however the projection-based algorithm in [3] converges faster (ref. Fig. 2a). A comparison of the time taken by Algorithm 1 and the projection algorithm is provided in Table II.

TABLE II: Average time taken (in seconds) by Algorithm 1 with $(\eta = 300, k = 900)$ and projection algorithm [3] for solving Example 2 over 5 runs.

Algorithm:	Algorithm 1	Projection algorithm
Time needed before iterations:	0.1775	0.4001
Time needed during iterations:	1.4169	0.6149
Total time needed:	1.5944	1.0150

V. DISCUSSION

In this paper, we proposed a continuous time algorithm to solve a system of linear algebraic equations $Ax = b$ distributively when there are delays in the communication links and the matrix A is ill-conditioned. Our algorithm utilizes preconditioners in conjunction with the classical distributed gradient method for optimization. A methodology for appropriately selecting the local preconditioner matrices has been provided so that the distributed gradient method globally converges to the desired solution of a set of equations $Ax = b$.

To illustrate the effectiveness of the proposed algorithm, we have applied it to solving a numerical example. It has been shown that the rate of convergence can be controlled by tuning the algorithm parameters, although no bound on convergence rate has been provided. From simulations, we have seen that the algorithm in [3] is faster than Algorithm 1. However, unlike the former algorithm, Algorithm 1 is guaranteed to converge without assuming upper bound on the delays τ^{ij}.

REFERENCES

[1] Peng Wang, Shaoshuai Mou, Jianming Lian, and Wei Ren. Solving a system of linear equations: From centralized to distributed algorithms. *Annual Reviews in Control*, 2019.

[2] Shaoshuai Mou, Ji Liu, and A Stephen Morse. A distributed algorithm for solving a linear algebraic equation. *IEEE Transactions on Automatic Control*, 60(11):2863–2878, 2015.

[3] Ji Liu, Shaoshuai Mou, and A Stephen Morse. Asynchronous distributed algorithms for solving linear algebraic equations. *IEEE Transactions on Automatic Control*, 63(2):372–385, 2017.

[4] Brian Anderson, Shaoshuai Mou, A Stephen Morse, and Uwe Helmke. Decentralized gradient algorithm for solution of a linear equation. *arXiv preprint arXiv:1509.04538*, 2015.

[5] S Sh Alaviani and Nicola Elia. A distributed algorithm for solving linear algebraic equations over random networks. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 83–88. IEEE, 2018.

[6] Tianyu Wu, Kun Yuan, Qing Ling, Wotao Yin, and Ali H Sayed. Decentralized consensus optimization with asynchrony and delays. *IEEE Transactions on Signal and Information Processing over Networks*, 4(2):293–307, 2017.

[7] Ji Liu, Xiaobin Gao, and Tamer Başar. A communication-efficient distributed algorithm for solving linear algebraic equations. In *2014 7th International Conference on NETwork Games, COntrol and OPtimization (NetGCoop)*, pages 62–69. IEEE, 2014.

[8] Xiaobin Gao, Ji Liu, and Tamer Başar. Stochastic communication-efficient distributed algorithms for solving linear algebraic equations. In *2016 IEEE Conference on Control Applications (CCA)*, pages 380–385. IEEE, 2016.

[9] Xuan Wang, Shaoshuai Mou, and Dengfeng Sun. Improvement of a distributed algorithm for solving linear equations. *IEEE Transactions on Industrial Electronics*, 64(4):3113–3117, 2016.

[10] Xuan Wang, Jingqiu Zhou, Shaoshuai Mou, and Martin J Corless. A distributed linear equation solver for least square solutions. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 5955–5960. IEEE, 2017.

[11] Yang Liu, Youcheng Lou, Brian Anderson, and Guodong Shi. Network flows that solve least squares for linear equations. *arXiv preprint arXiv:1808.04140*, 2018.

[12] John Tsitsiklis, Dimitri Bertsekas, and Michael Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE transactions on automatic control*, 31(9):803–812, 1986.

[13] Konstantinos I Tsianos and Michael G Rabbat. Distributed dual averaging for convex optimization under communication delays. In *2012 American Control Conference (ACC)*, pages 1067–1072. IEEE, 2012.

[14] Håkan Terelius, Ufuk Topcu, and Richard M Murray. Decentralized multi-agent optimization via dual decomposition. *IFAC proceedings volumes*, 44(1):11245–11251, 2011.

[15] Takeshi Hatanaka, Nikhil Chopra, Takayuki Ishizaki, and Na Li. Passivity-based distributed optimization with communication delays using PI consensus algorithm. *IEEE Transactions on Automatic Control*, 63(12):4421–4428, 2018.

[16] Tao Yang, Jemin George, Jiahu Qin, Xinlei Yi, and Junfeng Wu. Distributed finite-time least squares solver for network linear equations. *arXiv preprint arXiv:1810.00156*, 2018.
[17] Vassilis Kekatos and Georgios B Giannakis. Distributed robust power system state estimation. *IEEE Transactions on Power Systems*, 28(2):1617–1626, 2012.
[18] Aryan Mokhtari, Qing Ling, and Alejandro Ribeiro. Network newton distributed optimization methods. *IEEE Transactions on Signal Processing*, 65(1):146–161, 2016.
[19] Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.
[20] Joao P Hespanha. *Linear systems theory*. Princeton university press, 2018.

APPENDIX

A. Proof of Lemma 1

It is well-known, if $\mathbb{G}$ is connected then (2) is equivalent to the unconstrained optimization problem (ref. [17], [18])

$$\underset{x\in\mathbb{R}^n,\forall i\in[m]}{\textit{minimize}} \sum_{i=1}^{m} \frac{1}{2}\left\|A^i x - b^i\right\|^2 . \tag{13}$$

Now the objective cost in (13) can be written as

$$\sum_{i=1}^{m}\left\|A^i x - b^i\right\|^2 = \left\|\begin{bmatrix} A^1 x - b^1 \\ \vdots \\ A^m x - b^m \end{bmatrix}\right\|^2 = \|Ax-b\|^2 . \tag{14}$$

Therefore, problems (3) and (13) are equivalent to each other. Hence, by transitivity property of equivalence relation, problems (2) and (3) are equivalent to each other.

B. Proof of Lemma 2

For any $i \in [m]$,

$$\begin{aligned}
&(\lambda, v) \text{ is an Eigen-pair of } \bar{K}^i \\
\iff &(K^i+\eta I_n)^{-1}(K^i-\eta I_n)v = \lambda v \\
\iff &(K^i-\eta I_n)v = (K^i+\eta I_n)\lambda v \text{ [This implies} \\
&\lambda \neq 1\text{, otherwise } v=0 \text{ is an Eigenvector of } \bar{K}^i.\text{]} \\
\iff &K^i v = \frac{1+\lambda}{1-\lambda}\eta v \\
\iff &(\frac{1+\lambda}{1-\lambda}\eta, v) \text{ is an Eigen-pair of } K^i.
\end{aligned}$$

We have $K^i = k(N^i)^{-1}$, which can easily be verified by substituting this expression in the Lyapunov equation (6) which has a unique solution. Then,

$$\begin{aligned}
&\lambda[K^i] = \frac{k}{\lambda[N^i]} = \frac{k}{\lambda[(A^i)^T A^i] + |\mathcal{N}^i|} > 0 \\
\iff &\frac{1+\lambda}{1-\lambda}\eta > 0 \iff |\lambda| < 1.
\end{aligned}$$

Therefore, $\bar{K}^i$ is Schur and the claim follows.

C. Proof of Lemma 3

We begin with the necessary definitions (ref. [15]):

$$\bar{r}_{ij} = r_{ij} - x^*, \tag{15}$$
$$\bar{x}^i = x^i - x^*. \tag{16}$$
$$\bar{S}^i = (1/2)\left\|\bar{x}^i\right\|^2, \tag{17}$$
$$\begin{aligned} V^{ij}(t) =& \frac{1}{2}\int_{t-\tau^{ij}}^{t}\left\|\vec{s}_{ij}(y) - \frac{1}{\sqrt{2\eta}}\eta x^*\right\|^2 dy \\ &+ \frac{1}{2}\int_{t-\tau^{ji}}^{t}\left\|\vec{s}_{ji}(y) - \frac{1}{\sqrt{2\eta}}\eta x^*\right\|^2 dy, \end{aligned} \tag{18}$$
$$V = \sum_{i\in[m]}\bar{S}^i + \sum_{(i,j)\in\mathbb{E}} V^{ij}. \tag{19}$$

As the proof is long, we outline the steps as follows:

1) Define a suitable time-dependent Lyapunov function candidate V (ref. (19)) for the combined agent dynamics, where the overall Lyapunov function is contributed by storage functionals $\bar{S}^i$ for individual agents and V^{ij} for their links.
2) The time derivative of V is shown to be non-positive, and consensus is established between the agents using extension on LaSalle's principle.
3) Finally, asymptotic convergence is established by showing that the solution set of the time derivative of the Lyapunov function V being identically zero only consists of each agent asymptotically reaching the desired solution x^*.

From (12) and (16),

$$\dot{\bar{x}}^i = \dot{x}^i = v^i - K^i\phi^i(x^i). \tag{20}$$

From (17), $\bar{S}^i$ is "positive definite" and

$$\dot{\bar{S}}^i = (\bar{x}^i)^T v^i - (\bar{x}^i)^T K^i (A^i)^T A^i (x^i - x^*). \tag{21}$$

From (8), (15) and (16),

$$v^{ij} = K^i(\bar{r}_{ij} - \bar{x}^i),\; v^i = \sum_{j\in\mathcal{N}^i} K^i(\bar{r}_{ij} - \bar{x}^i).$$

Substituting them into (21),

$$\begin{aligned}
\dot{\bar{S}}^i =& (\bar{x}^i)^T\sum_{j\in\mathcal{N}^i} K^i(\bar{r}_{ij} - \bar{x}^i) - (\bar{x}^i)^T K^i (A^i)^T A^i \bar{x}^i \\
=& \sum_{j\in\mathcal{N}^i}[(\bar{r}_{ij})^T v^{ij} - (\bar{x}^i - \bar{r}_{ij})^T K^i(\bar{x}^i - \bar{r}_{ij})] \\
&- (\bar{x}^i)^T K^i (A^i)^T A^i \bar{x}^i.
\end{aligned} \tag{22}$$

We use the following facts (ref. proof of Lemma 5 in [15]):

$$V^{ij}(t) \geq 0 \quad \forall t \tag{23}$$
$$\text{and } \dot{v}^{ij}(t) = -(v^{ij})^T\bar{r}_{ij} - (v^{ji})^T\bar{r}_{ji}. \tag{24}$$

From (22) and (24) it follows that,

$$\begin{aligned}
\dot{V} =& -\sum_{i\in[m]}\sum_{j\in\mathcal{N}^i}(\bar{x}^i - \bar{r}_{ij})^T K^i (\bar{x}^i - \bar{r}_{ij}) \\
&- \sum_{i\in[m]}(\bar{x}^i)^T K^i (A^i)^T A^i \bar{x}^i \leq 0
\end{aligned} \tag{25}$$

So, $x^i \in \mathcal{L}_\infty \; \forall i$. From (8)-(11),

$$r_{ij}(t) = (\bar{K}^i)^2 r_{ij}(t - \tau^{ij} - \tau^{ji}) + \beta^{ij}(t), \tag{26}$$

where $\beta^{ij}(t)$ are linear functions of x^i and x^j at times t, $t-\tau^{ij}$, $t-\tau^{ji}$, $t-\tau^{ij}-\tau^{ji}$ (ref. [15]). Also $\beta^{ij}(t)$ are bounded linear, because $x^i \in \mathcal{L}_\infty$. Lemma 2 states that Eigenvalues of $\bar{K}^i \; \forall i$ are within unit circle, if $\eta > 0$. Then, (26) is a stable difference equation with bounded inputs, which means $r_{ij} \in \mathcal{L}_\infty$. So, extension of LaSalle's principle for time delay systems is applicable. Now,

$$\begin{aligned} \dot{V} \equiv 0 \iff \bar{x}^i = \bar{r}_{ij} \, \forall i, \forall j \in \mathcal{N}^i, \\ \sum_{i \in [m]} (\bar{x}^i)^T K^i (A^i)^T A^i \bar{x}^i = 0. \end{aligned} \tag{27}$$

Thus, LaSalle's principle implies that

$$\forall i, \, x^i \to r_{ij} \text{ as } t \to \infty \, \forall j \in \mathcal{N}^i. \tag{28}$$

From (8)-(11) one can see that,

$$\begin{aligned} \eta r_{ij}(t) =& \sqrt{2\eta}\vec{s}_{ji}(t - \tau^{ji}) - K^i(r_{ij}(t) - x^i(t)) \\ =& - K^j(r_{ji}(t - \tau^{ji}) - x^j(t - \tau^{ji})) \\ &+ \eta r_{ji}(t - \tau^{ji}) - K^i(r_{ij}(t) - x^i(t)). \end{aligned} \tag{29}$$

From (28) and (29), for every $i, \forall j \in \mathcal{N}^i$,

$$\begin{aligned} &\lim_{t\to\infty} r_{ij}(t) = \lim_{t\to\infty} r_{ji}(t - \tau^{ji}) = \lim_{t\to\infty} r_{ji}(t) \\ \implies &\lim_{t\to\infty} x^i(t) = \lim_{t\to\infty} x^j(t). \end{aligned}$$

Thus consensus is achieved asymptotically. Then, from (27) we have

$$\begin{aligned} \dot{V} \equiv 0 \implies &\lim_{t\to\infty} x^i(t) = \lim_{t\to\infty} x(t) \, \forall i, \\ &\lim_{t\to\infty} \sum_{i \in [m]} (\bar{x}(t))^T K^i (A^i)^T A^i \bar{x}(t) = 0. \end{aligned} \tag{30}$$

for some $x : \mathbb{R} \to \mathbb{R}^n$ and $\bar{x} := x - x^*$. So, the only thing left to be shown is

$$\lim_{t\to\infty} \sum_{i \in [m]} (\bar{x}(t))^T K^i (A^i)^T A^i \bar{x}(t) = 0 \implies \lim_{t\to\infty} \bar{x}(t) = 0.$$

So it is sufficient to show that,

$$\sum_{i \in [m]} K^i (A^i)^T A^i \succ 0. \tag{31}$$

We have

$$K^i \in S^n_{++}, (A^i)^T A^i \in S^n_{+} \, \forall i \implies \sum_{i \in [m]} K^i (A^i)^T A^i \succeq 0$$

For each i, consider the Eigen decomposition $(A^i)^T A^i = U^i \Lambda^i (U^i)^T$, where U^i is the Eigenvector matrix of $(A^i)^T A^i$ and Λ^i is a diagonal matrix with the Eigenvalues of $(A^i)^T A^i$ in the diagonal. Then,

$$N^i = |\mathcal{N}^i| I_n + (A^i)^T A^i = U^i (|\mathcal{N}^i| I_n + \Lambda^i)(U^i)^T.$$

We have $K^i = k(N^i)^{-1}$, which can easily be verified by substituting in the Lyapunov equation (6) which has a unique solution. Then,

$$K^i (A^i)^T A^i = k U^i diag\{\frac{\lambda_{ik}}{|\mathcal{N}^i| + \lambda_{ik}}\}_{k=1}^n (U^i)^T,$$

where λ_{ik} are the Eigenvalues of $(A^i)^T A^i \in S^n_+$, and $\lambda_{ik} = 0$ if the k^{th} column of U^i is in $null(A^i)$. This also means $K^i (A^i)^T A^i \in S^n_+$.

Consider any $x \in \mathbb{R}^n, x \neq 0$. Now $\exists j^* \in [m]$ s.t. $x \notin null(A^{j^*})$. Otherwise, $x \in null(A^i) \; \forall i$ which implies $x \in null(A)$. From Assumption 3, $null(A) = 0$ which is a contradiction. Now,

$$x^T K^{j^*} (A^{j^*})^T A^{j^*} x = k \sum_{k=1}^n \frac{\lambda_{j^*k}}{|\mathcal{N}^{j^*}| + \lambda_{j^*k}} \left\| x^T u_{j^*k} \right\|^2. \tag{32}$$

We consider the nontrivial case where for each i, A^i is not the zero matrix. Then, a positive Eigenvalue exists, because $(A^{j^*})^T A^{j^*} \in S^n_+$ and all of its Eigenvalues cannot be zero. Define $l = n - \eta((A^{j^*})^T A^{j^*})$. Then $\exists l \geq 1$ such that $\lambda_{j^*k} > 0, k = 1, ..., l$ (possibly by rearrangement of indices) and the Eigenvectors $\{u_{j^*k}\}_{k=1}^l \not\subset null((A^{j^*})^T A^{j^*})$ and $\{u_{j^*k}\}_{k=l+1}^n = null((A^{j^*})^T A^{j^*})$. Then,

$$\begin{aligned} x^T K^{j^*} (A^{j^*})^T A^{j^*} x = 0 &\iff x \perp span\{u_{j*k}\}_{k=1}^l \\ &\iff x \in span\{u_{j*k}\}_{k=l+1}^n \\ &\iff x \in null((A^{j^*})^T A^{j^*}) \end{aligned}$$

Therefore, $x^T K^{j^*} (A^{j^*})^T A^{j^*} x > 0$ and the claim follows from (31).

D. Proof of Theorem 1

For every $i \in [m]$, consider the system (12) with the definitions (9)-(11). Then, Lemma 3 implies that $x^i \to x^*$ $\forall i$ as $t \to \infty$. So it is enough to show that, there exists $\vec{s}_{ij} \; \forall j \in \mathcal{N}^i$ satisfying definitions (9)-(11) such that the dynamics in (7) is same as the dynamics (12).

Let

$$\vec{s}_{ij}(t) = \sqrt{\frac{\eta}{2}} x^i(t) \, \forall j \in \mathcal{N}^i. \tag{33}$$

From (11) and the scattering variables chosen as above,

$$\overleftarrow{s}_{ij}(t) = \vec{s}_{ji}(t - \tau^{ji}) = \sqrt{\frac{\eta}{2}} x^j(t - \tau^{ji}). \tag{34}$$

From (8), (9) and (34) it can be seen that,

$$\begin{aligned} &\eta x^j(t - \tau^{ji}) = K^i(r_{ij}(t) - x^i(t)) + \eta r_{ij}(t) \\ \implies &r_{ij}(t) = (K^i + \eta I_n)^{-1}(\eta x^j(t - \tau^{ji}) + K^i x^i(t)). \end{aligned}$$

By the above equation and (8),

$$\begin{aligned} v^{ij}(t) =& K^i[(K^i + \eta I_n)^{-1}(\eta x^j(t - \tau^{ji}) + K^i x^i(t)) - x^i(t)] \\ =& \eta K^i (K^i + \eta I_n)^{-1}(x^j(t - \tau^{ji}) - x^i(t)), \end{aligned} \tag{35}$$

where the last equality follow from the fact

$$\begin{aligned} &(K^i + \eta I_n)^{-1} K^i - I_n \\ =& (K^i + \eta I_n)^{-1} K^i - (K^i + \eta I_n)^{-1}(K^i + \eta I_n) \\ =& - \eta (K^i + \eta I_n)^{-1}. \end{aligned}$$

Equation (35) substituted in (12) leads to (7), and the choice of variables in (33) satisfy definitions (9)-(11). Therefore, the claim follows from Lemma 3.