5-2018

Consensus and Platooning in Multiagent Networks

Jiazhi Song

Purdue University

CONSENSUS AND PLATOONING IN MULTIAGENT NETWORKS

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Jiazhi Song

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science in Aeronautics and Astronautics

May 2018

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF DISSERTATION APPROVAL

Dr. Shaoshuai Mou, Chair

School of Aeronautics and Astronautics

Dr. Dengfeng Sun

School of Aeronautics and Astronautics

Dr. Shreyas Sundaram

School of Aeronautics and Astronautics

Approved by:

Dr. Weinong Chen

Head of the School Graduate Program

Dedicated to the memory of my grandfather Shulin Song (1943-2017).

ACKNOWLEDGMENTS

Foremost, I would like to express my sincere gratitude to my advisor Prof. Shaoshuai Mou for the continuous support of my Master's study and research. His guidance helped me during research and in writing of this thesis. Besides my advisor, I would like to thank Prof. Dengfeng Sun, Prof. Shreyas Sundarum, and Prof. Martin Corless for their help and kindness. I also would like to thank all my labmates and friends at Purdue for their encouragements and companionship. Last but not the least, I would like to thank members of my family for their financial and emotional support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

ACA	Accelerated Consensus-based Algorithm
ACC	Adaptive Cruise Control
CACC	Adaptive Cruise Control
DALE	Distributed Algorithm for solving Linear Equation

ABSTRACT

Song, Jiazhi MSAA, Purdue University, May 2018. Consensus and Platooning in Multiagent Networks. Major Professor: Shaoshuai Mou.

First, a distributed algorithm to accelerate the convergence of a class of linear time-invariant consensus-based distributed algorithms is proposed. Then, it is proven that, given a convergent distributed algorithm, the acceleration algorithm can ensure convergence and consensus. Also, the parameter that can ensure the largest possible convergence speed was determined. Furthermore, it is shown that some constraints on the equilibrium state of the original algorithms also apply to the equilibrium state of the acceleration algorithm. Finally, some examples are presented to validate the effectiveness of the acceleration algorithm. A method that allows obtaining convergence value within a finite amount of time is also discussed.

Then, this paper studies the longitudinal string stability of two cooperative adaptive cruise-control(CACC) equipped 2-vehicle platoons implementing different inter-platoon communication topologies. CACC utilizes wireless communication between vehicles to improve the performance of the tested and commercialized adaptive cruise control(ACC). Due to 2-vehicle CACC platoon being well studied and tested, inter-platoon communication is used to connect multiple 2-vehicle platoons and therefore accommodate more vehicles to form a larger platoon for better energy saving. Frequency domain approach is used to carry out string stability analysis. A general form of feedforward filter was derived and different inter-platoon communication topologies are analytically proven to be string stable under delay-free environment. The minimum headway time of each communication topology is then presented to show the effect of communication structure and delay on string stability.

1. ACCELERATION OF CONSENSUS-BASED DISTRIBUTED ALGORITHM

1.1 Introduction

Distributed multi-agent system is a popular topic in the research community due to its potential in network security and efficiency. A distributed multi-agent system is established upon a network in which each agent can communicate with each other by sending its own states and receiving other agents' states. The system is distributed because the presence of a central agent is not required so that all the agents can work together while having the same level of capability [1].

For an algorithm that allows a group of agents to reach a common goal through communication, as in [2–14] it usually involves a iterative update that involves each agent's and its neighbors' latest states. For the state of each agent to reach consensus, the convergence rate are usually exponential. Therefore, a lot of previous research effort [11, 13, 15–22] was spent on speeding up the convergence rate of distributed algorithms. Among them, [15–17] worked on speeding up original algorithms using states from previous steps, and [11, 13, 18–22] worked on finding ways to ensure the convergence of distributed algorithms within finite time.

The method used in this work is inspired by the well-known Successive Over-Relaxation (SOR) method [23, 24] that was developed for the acceleration of centralized computing. Although achieving the acceleration of a distributed algorithm using a method that is over-relaxation inspired as in [15–17], this work is focused on a more general constrained consensus problem while the others focused on distributed averaging. The more general constrained consensus problems considered in this work include distributed algorithm for solving linear equations as introduced in [25] and one cannot simply take the results achieved in [15–17] and apply. Among the works that addressed finite-time convergence, [11, 21] required specific graph structures, and [13, 18–20] only considered distributed averaging.

This study illustrates how a new distributed Accelerated Consensus-based Algorithm (ACA) can speed up general consensus-based distributed algorithms by using an additional memory space of each agent. A way of selecting the best algorithm design parameter is also introduced. This study also introduces a more general finite-time solution method that is capable of reaching the convergence value of original algorithms within finite time steps. Unlike the previous research works, the algorithms introduced in this work can be applied to a class of consensus-based distributed algorithms including distributed consensus, distributed averaging, and distributed algorithm for solving linear equation. Also, they do not require the communication topology to have a special structure.

1.2 Problem Formulation

Consider a connected undirected graph $\mathbb{G}$ consisting m vertices and p edges where each vertex stands for an agent and each edge stands for a communication link between the agents on each end. When two agents are connected by an communication link, they are called neighbors of each other and the set of neighbors of agent i is denoted by $\mathcal{N}_i$ for $i = 1, \ldots, m$. Each agent has its own state $x_i \in \mathbb{R}^n$. A lot of discrete linear distributed consensus-based algorithms constructed on communication graph like $\mathbb{G}$ have the form:

$$x_i(t+1) = \sum_{j \in \mathcal{N}_i} W_{ij} x_j(t), \tag{1.1}$$

with t representing time index. Due to the algorithms being linear, (1.1) can be written in the systems matrix form

$$\boldsymbol{x}(t+1) = W \boldsymbol{x}(t), \tag{1.2}$$

where

$$\boldsymbol{x}(t) = \begin{bmatrix} x_1(t) \\ \vdots \\ x_m(t) \end{bmatrix},$$

and W with dimension $mn \times mn$ is determined by the network. Since the nature of consensus problem is that the state of each agent eventually converge to the same, we can write the equilibrium state of (1.2) as

$$\boldsymbol{x}_e = \left[\mathbf{1}_m \otimes x^*\right] \tag{1.3}$$

where $\otimes$ stands for kronecker product, $\mathbf{1}_m$ is a m-length vector with all entries equal to 1, and x^* can be any n-length vector. The algorithms like (1.1) usually allow each agent's state to converge to x^* as fast as $\rho^t \to 0$ as $t \to \infty$ with $\rho \in \mathbb{R}$ and $0 < \rho < 1$. Here ρ is the exponential convergence rate of (1.1) and is determined by the largest magnitude of system matrix W's eigenvalues that is not 1. For an algorithm like (1.1), whose system matrix W's eigenvalues are all real, this work proposes to introduce a method to accelerate the convergence so that the new convergence rate $\bar{\rho}$ satisfies $0 < \bar{\rho} < \rho$.

Some algorithms that motivated the development of the proposed method include the distributed algorithm for solving linear equations introduced in [25]. It solves a linear equation of the form $Ax^* = b$ with dimension n using m agents. Each agent has the knowledge of a part of the augmented matrix $[A, b]$ denoted by $[A_i, b_i]$. It has the form

$$x_i(t+1) = x_i(t) - \frac{1}{d_i} P_i \left(d_i x_i(t) - \sum_{j \in \mathcal{N}_i} x_j(t) \right), \tag{1.4}$$

where d_i stands for the amount of neighbors of agent i, and P_i is the readily computable orthogonal projection on the kernel of A_i.

Notice that (1.4) ensures that the convergence value of each agent's state satisfy the constraint $Ax^* = b$, the proposed method also tries to ensure the convergence value satisfies such linear constraint.

1.3 Accelerated Consensus-based Algorithm

1.3.1 The update

Under the assumption that the original update is of the form (1.1) and can be written as (1.2), we propose an SOR-inspired Accelerated Consensus Algorithm (ACA) as follows with the expectation to accelerate convergence:

$$x_i(t+1)=\begin{cases}\sum_{j\in\mathcal{N}_i} W_{ij}x_j(t), & \text{if } t=0\\ \alpha\sum_{j\in\mathcal{N}_i} W_{ij}x_j(t)+(1-\alpha)x_i(t-1), & \text{if } t\geq 1\end{cases}. \tag{1.5}$$

Note that (1.5) only requires each agent to store an additional step of state and does not require any other additional information. It is also easy to see that (1.5) can be written in system matrix form as

$$\boldsymbol{x}(t+1)=\begin{cases}W\boldsymbol{x}(t), & \text{if } t=0\\ \alpha W\boldsymbol{x}(t)+(1-\alpha)\boldsymbol{x}(t-1), & \text{if } t\geq 1\end{cases}. \tag{1.6}$$

1.3.2 Analysis

It is easy to illustrate that any linear constraint on the state of a original algorithm still holds when ACA is utilized using proof by induction. First, consider the original system update matrix W satisfies $CW = C$ and results in the relation $CW^t\boldsymbol{x}(0) = C\boldsymbol{x}(0)$, where C is a matrix of appropriate dimension that stands for a linear constraint. For (1.6), we know that $C\boldsymbol{x}(1) = C\boldsymbol{x}(0)$ is true because (1.6) is equivalent to the original update for $t = 0$. Then, for $t = 1$, we get

$$\begin{aligned}C\boldsymbol{x}(2) &= \alpha CW\boldsymbol{x}(1) + (1-\alpha)\boldsymbol{x}(0)\\ &= \alpha C\boldsymbol{x}(1) + (1-\alpha)C\boldsymbol{x}(1)\\ &= C\boldsymbol{x}(1).\end{aligned} \tag{1.7}$$

Then, suppose we have $C\boldsymbol{x}(t) = C\boldsymbol{x}(t-1)$, and $CW = C$ for $t \geq 1$, we get that

$$\begin{aligned} C\boldsymbol{x}(t+1) &= \alpha CW\boldsymbol{x}(t) + (1-\alpha)C\boldsymbol{x}(t-1) \\ &= \alpha C\boldsymbol{x}(t) + (1-\alpha)C\boldsymbol{x}(t) \\ &= C\boldsymbol{x}(t). \end{aligned}$$

Considering we already have $C\boldsymbol{x}(2) = C\boldsymbol{x}(1)$ from (1.7) and $CW = C$, we know $C\boldsymbol{x}(t+1) = C\boldsymbol{x}(t)$ is true for $t \geq 1$ using proof by induction. Hence, considering we already have $C\boldsymbol{x}(1) = C\boldsymbol{x}(0)$, we know that $CW^t\boldsymbol{x}(0) = C\boldsymbol{x}(0)$ is true for all $t \geq 0$ and we know that any constraint of the form still holds when ACA is utilized.

Considering for (1.4), the lower bound of ρ satisfies $0 < \tilde{\rho} < \rho$, and all eigenvalues of W are real [25], the main result of this work is as follows

Theorem 1. *For the original update (1.2) whose convergence rate ρ has lower bound $\tilde{\rho}$, the state of (1.6) reaches consensus (1.3) at least as fast as $\bar{\rho}^t \to 0$ as $t \to \infty$ with $0 < \bar{\rho} < \rho$ for $\alpha \in (1, \frac{2-2\sqrt{1-\tilde{\rho}^2}}{\tilde{\rho}^2}]$.*

To prove the theorem, we first arrange the accelerated consensus algorithm (1.6) into an augmented matrix form

$$\bar{\boldsymbol{x}}(t+1) = \bar{W}\bar{\boldsymbol{x}}(t), \tag{1.8}$$

where

$$\bar{W} = \begin{bmatrix} \alpha W & (1-\alpha)I_{mn} \\ I_{mn} & 0 \end{bmatrix}, \bar{\boldsymbol{x}}(t) = \begin{bmatrix} \boldsymbol{x}(t) \\ \boldsymbol{x}(t-1) \end{bmatrix}$$

for $t \geq 1$. Here I_{mn} stands for a identity matrix whose dimension is indicated by its subscript, in this case, mn. Because the convergence of $\boldsymbol{x}(t) \to \boldsymbol{x}_e$ is equivalent to

$$\bar{\boldsymbol{x}}_e \to \begin{bmatrix} \boldsymbol{x}_e \\ \boldsymbol{x}_e \end{bmatrix},$$

the convergence properties of (1.6) is naturally equivalent to that of matrix $\bar{W}$.

To further prove the convergence of (1.6), we introduce the following lemma.

Lemma 1. *Suppose all eigenvalues of W are real with n nondefective eigenvalues equal to* 1 *and others have magnitude strictly less than* 1*. $v_1, \cdots, v_n$, the eigenvectors correspond to* 1*-eigenvalues, are of the form $\mathbf{1}_m \otimes r_i$ with $i = 1, \cdots, n$ and r_i's are n-vectors that span $\mathbb{R}^n$.*

Lemma 1 is a summary of the results from [5] and [25], interested readers are directed to these two papers.

Define λ as any eigenvalue of W, $\bar{\lambda}$ as any eigenvalue of $\bar{W}$, and a nonzero vector

$$\begin{bmatrix} u \\ w \end{bmatrix}$$

where $u, w \in \mathbb{R}^{mn}$. According to matrix properties, the following equation holds for all eigenvalues $\bar{\lambda}$ of $\bar{W}$:

$$\begin{bmatrix} \alpha W & (1-\alpha) I_{mn} \\ I_{mn} & 0 \end{bmatrix} \begin{bmatrix} u \\ w \end{bmatrix} = \bar{\lambda} \begin{bmatrix} u \\ w \end{bmatrix}. \tag{1.9}$$

From (1.9), we can get the following

$$\alpha W u + (1-\alpha) w = \bar{\lambda} u \tag{1.10}$$

$$u = \bar{\lambda} w. \tag{1.11}$$

From (1.11), since $\bar{\lambda}$ is a scalar and u,w cannot both be zero, it is obvious that vector $w \neq 0$. Substitute (1.11) into (1.10), we get

$$Ww = \frac{\bar{\lambda}^2 - 1 + \alpha}{\alpha \bar{\lambda}} w.$$

Due to w being non-zero, we get

$$\lambda = \frac{\bar{\lambda}^2 - 1 + \alpha}{\alpha \bar{\lambda}}$$

is an eigenvalue of W. Therefore, all nonzero eigenvalues of $\bar{W}$ can be expressed by λ as

$$\bar{\lambda} = \frac{\alpha\lambda \pm \sqrt{\alpha^2\lambda^2 + 4 - 4\alpha}}{2}. \tag{1.12}$$

Recall that $\tilde{\rho}$ is the lower bound of ρ, where ρ is the convergence rate of (1.2) as introduced above, the following lemma is introduced.

Lemma 2. *For the algorithm as shown in (1.8), $\bar{\rho}$, the largest magnitude of $\bar{W}$'s eigenvalues that are not 1, satisfies the relation $\bar{\rho} < \rho$ for $\alpha \in (1, \frac{2-2\sqrt{1-\tilde{\rho}^2}}{\tilde{\rho}^2}]$, and $\bar{\rho}$ is at its minimum when $\alpha = \frac{2-2\sqrt{1-\tilde{\rho}^2}}{\tilde{\rho}^2}$.*

Proof of Lemma 2. To find out how λ affects the magnitude of $\bar{\lambda}$ for $1 < \alpha < 2$, we first consider the case where $\bar{\lambda}$ is a real number and denote it as $|\bar{\lambda}^{\mathbb{R}}|$. For this case, $\alpha^2\lambda^2 + 4 - 4\alpha \geq 0$, and λ cannot be 0. Denote $\lambda > 0$ as λ^+ and $\lambda < 0$ as λ^-. Since $\bar{\lambda}$ is real, for any given λ, the larger magnitude of (1.12) is either given by

$$|\bar{\lambda}^+| = \frac{\alpha\lambda^+ + \sqrt{\alpha^2\lambda^{+^2} + 4 - 4\alpha}}{2} \tag{1.13}$$

or

$$|\bar{\lambda}^-| = \frac{-\alpha\lambda^- + \sqrt{\alpha^2\lambda^{-^2} + 4 - 4\alpha}}{2}. \tag{1.14}$$

Here, $\bar{\lambda}^+$ is the eigenvalue of $\bar{W}$ that corresponds to λ^+, and $\bar{\lambda}^-$ is the eigenvalue of $\bar{W}$ that corresponds to λ^-. To find out how different $|\lambda|$ values affect the largest $|\bar{\lambda}^{\mathbb{R}}|$, we take derivative of (1.13) and (1.14) with respect to $|\lambda|$ and get

$$\frac{\partial|\bar{\lambda}^{\mathbb{R}}|}{\partial|\lambda|} > 0. \tag{1.15}$$

For the case where $\bar{\lambda}$ is complex, we denote it as $\bar{\lambda}^{\mathbb{C}}$. And $\bar{\lambda}$ being complex is equivalent to $\alpha^2\lambda^2 + 4 - 4\alpha < 0$. So

$$|\bar{\lambda}^{\mathbb{C}}| = \sqrt{\alpha - 1}. \tag{1.16}$$

From (1.12), we can observe that $|\bar{\lambda}^{\mathbb{R}}| = 1$ is true only when $|\lambda| = 1$. When $|\lambda| < 1$, considering $1 < \alpha < 2$, we can derive the following

$$\begin{aligned}
4\alpha - 4 &> (4\alpha - 4)\lambda^2 \\
\frac{\alpha^2\lambda^2 + 4 - 4\alpha}{4} &< \frac{\alpha^2 + 4 - 4\alpha}{4}\lambda^2 \\
\frac{\sqrt{\alpha^2\lambda^2 + 4 - 4\alpha}}{2} &< \frac{2-\alpha}{2}|\lambda| \\
\frac{\alpha|\lambda| + \sqrt{\alpha^2\lambda^2 + 4 - 4\alpha}}{2} &< |\lambda| \\
|\bar{\lambda}^{\mathbb{R}}| &< |\lambda|.
\end{aligned} \tag{1.17}$$

Because $\bar{\lambda}^{\mathbb{R}}$ exists is equivalent to $\alpha^2\lambda^2 + 4 - 4\alpha \geq 0$, we know it is also equivalent to

$$|\lambda| \geq \frac{2\sqrt{\alpha - 1}}{\alpha}. \tag{1.18}$$

Since it has been shown by (1.15) that $|\bar{\lambda}^{\mathbb{R}}|$ increases with $|\lambda|$, substituting (1.18) into (1.12), we get

$$|\bar{\lambda}^{\mathbb{R}}| \geq \sqrt{\alpha - 1}. \tag{1.19}$$

Hence,

$$|\bar{\lambda}^{\mathbb{R}}| \geq |\bar{\lambda}^{\mathbb{C}}|. \tag{1.20}$$

Because, from (1.17) and (1.20), $|\lambda| > |\bar{\lambda}^{\mathbb{R}}| \geq |\bar{\lambda}^{\mathbb{C}}|$ holds true for each λ that is not 1 when $|\bar{\lambda}^{\mathbb{R}}|$ exists, assuming $\tilde{\rho}$ is known, we require

$$\alpha^2\tilde{\rho}^2 + 4 - 4\alpha \geq 0. \tag{1.21}$$

After solving the inequality (1.21), we get

$$\alpha \geq \frac{2 + 2\sqrt{1 - \tilde{\rho}^2}}{\tilde{\rho}^2} \tag{1.22}$$

$$\alpha \leq \frac{2 - 2\sqrt{1 - \tilde{\rho}^2}}{\tilde{\rho}^2}. \tag{1.23}$$

It can be verified that (1.22) does not satisfy $1 < \alpha < 2$, while (1.23) always satisfies the relation

$$1 < \frac{2 - 2\sqrt{1 - \tilde{\rho}^2}}{\tilde{\rho}^2} < 2$$

for $0 < \tilde{\rho} < 1$. Therefore, the interval of α to guarantee $\bar{\rho} < \rho$ is

$$1 < \alpha \leq \frac{2 - 2\sqrt{1 - \tilde{\rho}^2}}{\tilde{\rho}^2}. \tag{1.24}$$

To decrease $\bar{\rho}$ as much as possible, we look at how $|\bar{\lambda}^{\mathbb{R}}|$ changes with α using

$$\frac{\partial|\bar{\lambda}^{\mathbb{R}}|}{\partial\alpha} = \frac{|\lambda|}{2} + \frac{2\alpha\lambda^2 - 4}{4\sqrt{\alpha^2\lambda^2 + 4 - 4\alpha}}. \tag{1.25}$$

We know (1.25) being negative is equivalent to

$$\begin{aligned} 2|\lambda|\sqrt{\alpha^2\lambda^2 + 4 - 4\alpha} + 2\alpha\lambda^2 - 4 < 0 \\ 4\lambda^2(\alpha^2\lambda^2 + 4 - 4\alpha) < 4\alpha^2\lambda^4 - 16\alpha\lambda^2 + 16 \\ 16\lambda^2 < 16. \end{aligned}$$

Since we are excluding the case $\lambda = 1$, we know it is true that $\frac{\partial|\bar{\lambda}^{\mathbb{R}}|}{\partial\alpha} < 0$ is true for all $\lambda \neq 1$. So $\alpha = \frac{2-2\sqrt{1-\bar{\rho}^2}}{\bar{\rho}^2}$ gives a minimized $\bar{\rho}$ that guarantees the relation $\bar{\rho} < \rho$. ■

With the introduced lemma, the convergence of ACA to consensus and its convergence speed can be shown.

Proof of Theorem 1. It can be observed from (1.12) that there is $\bar{\lambda} = 1$ when $\lambda = 1$. Combine this observation with Lemma 1 and Lemma 2, we know that there are n real eigenvalues of $\bar{W}$ equal to 1 and all the other eigenvalues of $\bar{W}$ have magnitude less than 1 with the largest being $\bar{\rho}$.

Then, there exists a nonzero vector

$$\begin{bmatrix} u_i \\ w_i \end{bmatrix}$$

for $i = 1, \cdots, n$ that satisfies

$$\begin{bmatrix} \alpha W & (1-\alpha)I_{mn} \\ I_{mn} & 0 \end{bmatrix} \begin{bmatrix} u_i \\ w_i \end{bmatrix} = \begin{bmatrix} u_i \\ w_i \end{bmatrix}. \tag{1.26}$$

Expanding (1.26), we get

$$\alpha W u_i + (1-\alpha) w_i = u_i \tag{1.27}$$

$$u_i = w_i. \tag{1.28}$$

Substituting (1.28) into (1.26), we get

$$\begin{bmatrix} \alpha W & (1-\alpha)I_{mn} \\ I_{mn} & 0 \end{bmatrix} \begin{bmatrix} u_i \\ u_i \end{bmatrix} = \begin{bmatrix} u_i \\ u_i \end{bmatrix}, \tag{1.29}$$

and substituting (1.28) into (1.27), we get

$$W u_i = u_i. \tag{1.30}$$

Then, it is shown by (1.30) that $\text{span}\{u_1, \cdots, u_n\} \in \text{span}\{v_1, \cdots, v_n\}$. Moreover, since it is shown by (1.29) that there should be n linearly independent u_i's, with the result of Lemma 1, we know the eigenvectors of $\bar{W}$ that corresponds to 1 eigenvalues are equivalent to

$$\begin{bmatrix} v_i \\ v_i \end{bmatrix}.$$

Hence we get $\bar{\boldsymbol{x}}(t)$ for $t \to \infty$ is

$$\lim_{t\to\infty} \bar{\boldsymbol{x}}(t) = \lim_{t\to\infty} U \begin{bmatrix} I_n & 0 \\ 0 & \Lambda^t \end{bmatrix} U^{-1} \bar{\boldsymbol{x}}(0), \tag{1.31}$$

where $U = \mathbf{1}_2 \otimes [u_1, \cdots, u_{2mn}]$, U^{-1} stands for the inverse of matrix U, and Λ is a block diagonal matrix whose entries have magnitude less than 1. Define β as a column vector of length n, it is then obtained from (1.31) that

$$\lim_{t\to\infty} \bar{\boldsymbol{x}}(t) = \mathbf{1}_2 \otimes \left(\begin{bmatrix} v_1 & \cdots & v_n \end{bmatrix} \beta \right), \tag{1.32}$$

and according to Lemma 1, (1.32) is equivalent to

$$\lim_{t\to\infty} \bar{\boldsymbol{x}}(t) = \mathbf{1}_2 \otimes (\mathbf{1}_m \otimes \bar{x}^*) \tag{1.33}$$

$$= \mathbf{1}_{2m} \otimes \bar{x}^*. \tag{1.34}$$

Equation (1.31) and (1.34) shows that ACA reaches consensus as fast as Λ^t converges to 0 as $t \to \infty$. Considering the convergence rate of Λ is no larger than its entry that has the largest magnitude, we get the rate of convergence to consensus of $\bar{\boldsymbol{x}}(t)$ as $t \to \infty$ is at least as fast as $\bar{\rho}$ where $0 < \bar{\rho} < \rho$. Note that $\bar{x}^*$ can be any n-length vector and does not have to equal x^* as the only requirement is that the state of each agent converges to be the same. ■

A special case for the convergence rate of the original algorithm is $\rho = 0$. For this case, we let $\alpha = 1$ and ACA eventually becomes the same as original algorithm and $\bar{\rho} = 0$ as well.

1.3.3 Example

Accelerated DALE

To demonstrate the effect of ACA, the following example with the application of ACA to DALE is shown. For a distributed algorithm that solves linear equations of

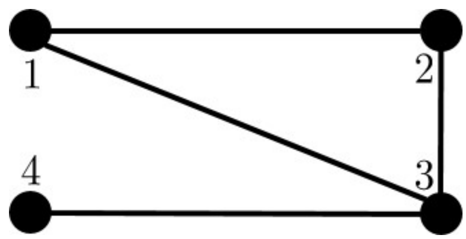


Figure 1.1.. A network of five agents connected by a undirected graph.

the form $Ax = b$ for x as introduced in [25], we illustrate the acceleration of DALE with a simulation. We consider four agents with undirected communication link as shown in Fig. 1.1 solve a randomly generated linear equation where

$$A=\begin{bmatrix} -0.301 & -0.206 & 0.306 & -0.244 & 0.225 \\ 0.005 & -0.241 & -0.408 & -0.396 & -0.100 \\ -0.401 & 0.298 & -0.789 & 0.248 & 0.200 \\ 0.154 & 0.310 & -0.438 & -0.519 & -0.194 \\ -0.666 & -0.483 & -0.167 & -0.029 & -0.400 \end{bmatrix}, b=\begin{bmatrix} 0.648 \\ -0.281 \\ 0.444 \\ -0.265 \\ -0.192 \end{bmatrix}. \quad (1.35)$$

Agent 1 knows the first line of $[A, b]$, agent 2 knows the second and third line of $[A, b]$, agent 3 knows the fourth line of $[A, b]$, and agent 4 knows the fifth line of $[A, b]$. Hence, DALE states that

$$H = \begin{bmatrix} 1 & 0 & 1 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & -1 & -1 & 1 \\ 0 & 0 & 0 & -1 \end{bmatrix},$$

$$D = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

And $W = I_{mn} - P\bar{D}^{-1}\bar{H}\bar{H}'$ has convergence rate 0.984 and assuming that we know its lower bound being $\tilde{\rho} = 0.96$. Then, according to Lemma 2, we set $\alpha = 1.563$. The Error vs. Iteration comparison between the original DALE and accelerated DALE is shown in Fig. 1.2 where the error is characterized by $\frac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}^e\|^2$. The accelerated DALE has a convergence rate of $\bar{\rho} = 0.938$.

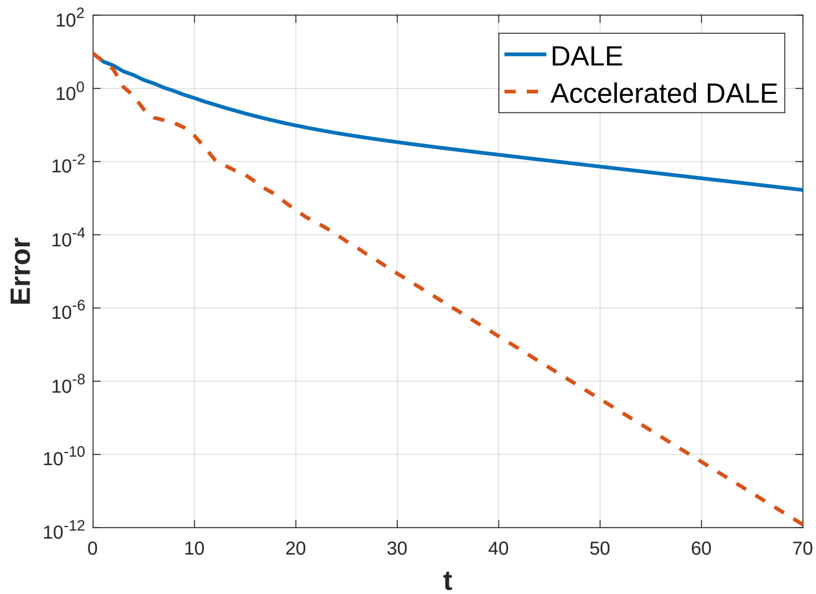


Figure 1.2.. Error vs. Iteration comparison between DALE and accelerated DALE.

Accelerated Distributed Averaging

For demonstration purposes, we also consider a distributed averaging problem for a undirected line graph that contains three agents with agent-3 connected to both

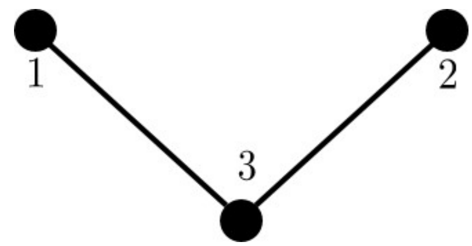


Figure 1.3.. A network of four agents connected by a undirected graph.

agent-1 and agent-2 as shown in Figure 1.3. Each agent has a state vector of length 2. The initial state of all three agents is a randomly generated vector

$$\boldsymbol{x}(0) = \begin{bmatrix} 15.238 \\ 82.582 \\ 53.834 \\ 99.614 \\ 7.818 \\ 44.268 \end{bmatrix}, \tag{1.36}$$

and the system update matrix is a symmetric doubly-stochastic matrix

$$W = \begin{bmatrix} \frac{2}{3} & 0 & 0 & 0 & \frac{1}{3} & 0 \\ 0 & \frac{2}{3} & 0 & 0 & 0 & \frac{1}{3} \\ 0 & 0 & \frac{1}{3} & 0 & \frac{2}{3} & 0 \\ 0 & 0 & 0 & \frac{1}{3} & 0 & \frac{2}{3} \\ \frac{1}{3} & 0 & \frac{2}{3} & 0 & 0 & 0 \\ 0 & \frac{1}{3} & 0 & \frac{2}{3} & 0 & 0 \end{bmatrix}.$$

We chose a symmetric W so that its eigenvalues are all real. The exact convergence rate of W is 0.578 and assuming that we know its lower bound being $\tilde{\rho} = 0.5$. Then, according to Lemma 2, we set $\alpha = 1.0718$. The Error vs. Iteration comparison between the original DALE and accelerated DALE is shown in Fig. 1.4 where the error is characterized by $\frac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}^e\|^2$. The accelerated DALE has a convergence rate of $\bar{\rho} = 0.4641$.

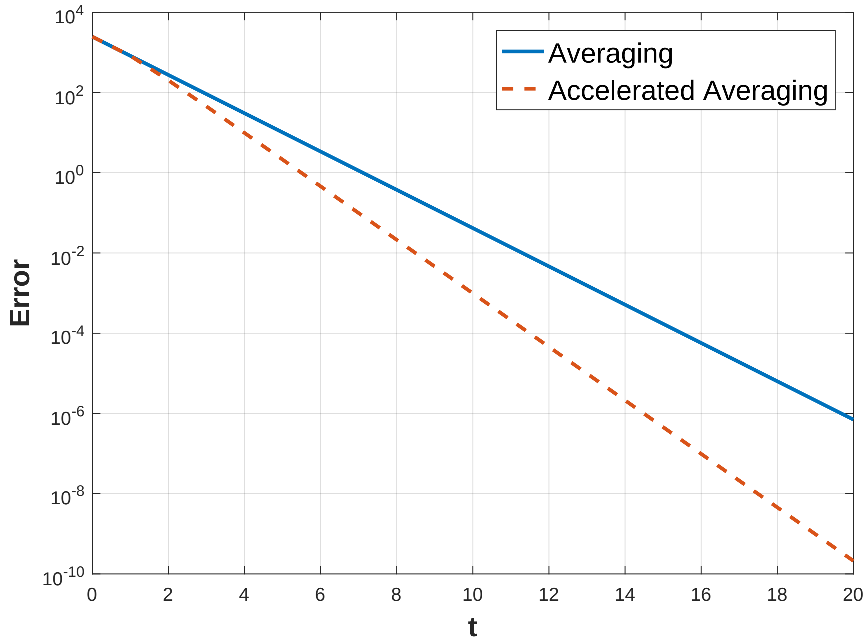


Figure 1.4.. Error vs. Iteration comparison between Distributed Averaging and accelerated Distributed Averaging.

1.4 Finite Time Solution

Aside from using a one-step storage space to speed up convergence, we can also take advantage of a system where each agent has large storage space and computing power to allow the convergence complete within finite time.

1.4.1 The Update

Under the same condition as mentioned above, if the system update matrix W satisfies the relation

$$\gamma_{mn}W^{mn} = -\gamma_{mn-1}W^{mn-1} - \gamma_{mn-2}W^{mn-2} - \cdots - \gamma_0 W^0,$$

the convergence value of (1.2) can be calculated by

$$\lim_{k\to\infty} x_i(k) = \frac{\left[x_i(0)\ x_i(1)\ \cdots\ x_i(mn-n)\right]\tilde{S}\tilde{h}_2}{S\tilde{h}_1}, \tag{1.37}$$

where $\tilde{S}$ is a upper anti-triangular Hankel matrix of the form

$$\tilde{S} = \begin{bmatrix} \sum_{i=n}^{mn} \gamma_i & \sum_{i=n+1}^{mn} \gamma_i & \cdots & \sum_{i=mn}^{mn} \gamma_i \\ \sum_{i=n+1}^{mn} \gamma_i & & \iddots & 0 \\ \vdots & \iddots & \iddots & \vdots \\ \sum_{i=mn}^{mn} \gamma_i & 0 & \cdots & 0 \end{bmatrix},$$

$$S = \begin{bmatrix} \sum_{i=n}^{mn} \gamma_i & \sum_{i=n+1}^{mn} \gamma_i & \cdots & \sum_{i=mn}^{mn} \gamma_i \end{bmatrix},$$

$$\tilde{h}_1 = \begin{bmatrix} h_{n-1}(1) \\ h_{n-1}(2) \\ \vdots \\ h_{n-1}(mn-n+1) \end{bmatrix}, \tag{1.38}$$

and

$$\tilde{h}_2 = \begin{bmatrix} h_{n-2}(1) \\ h_{n-2}(2) \\ \vdots \\ h_{n-2}(mn-n+1) \end{bmatrix}. \tag{1.39}$$

When $n = 1$, $\tilde{h}_2$'s first element equal to 1 and all others equal to 0. The variable $h_b(a)$ appeared in (1.38) and (1.39) stands for the a-th value in the b-th diagonal of a Pascal's triangle and can be calculated using the following equality:

$$h_b(a) = \binom{a+b-1}{b} = \frac{(a+b-1)!}{b!(a-1)!}. \tag{1.40}$$

The distributed method to determine γ_i's and the derivation of (1.37) will be introduced in the sequel.

1.4.2 Analysis

According to Cayley-Hamilton theorem, an $mn \times mn$ matrix W of (1.2) satisfies the relation $p(W) = 0$ where $p(\lambda) = det(\lambda I_{mn} - W)$ is the characteristic polynomial of W. Here I_{mn} is a $mn \times mn$ identity matrix and λ is a variable. Then $p(W)$ can be written as

$$W^{mn} = -\gamma_{mn-1} W^{mn-1} - \gamma_{mn-2} W^{mn-2} - \cdots - \gamma_0 W^0. \tag{1.41}$$

Each γ_j is a scalar coefficient of the characteristic equation (1.41). Multiply both sides of the equation by $\boldsymbol{x}(t)$, we get

$$\begin{aligned} \boldsymbol{x}(t+mn) = &- \gamma_{mn-1}\boldsymbol{x}(t+mn-1) - \gamma_{mn-2}\boldsymbol{x}(t+mn-2) \\ &- \cdots - \gamma_0 \boldsymbol{x}(t) \qquad \text{for } t \geq 0. \end{aligned} \tag{1.42}$$

Recall that $\boldsymbol{x}$ is constructed as in (1.2) and γ_j's are scalars, each row of (1.42) also satisfies its own equality. Therefore, it is obvious that the following equality also holds:

$$\begin{aligned} x_i(t+mn) = &- \gamma_{mn-1} x_i(t+mn-1) - \gamma_{mn-2} x_i(t+mn-2) \\ &- \cdots - \gamma_0 x_i(t) \qquad \text{for } t \geq 0. \end{aligned} \tag{1.43}$$

A state update of the form (1.43) is called a independent update due to the fact that it allows a single agent to perform iterative update of its state using the polynomial instead of communicating with others. Considering it has been proven that (1.43) exists for any agent in system (1.2) after the mn-th step, it is obvious that the independent update polynomial can be written in a matrix form

$$X_i \hat{\gamma} = \hat{X}_i \tag{1.44}$$

where

$$X_i = \begin{bmatrix} x_i(t) & \cdots & x_i(t+mn-1) \\ x_i(t+1) & \cdots & x_i(t+mn) \\ \vdots & \vdots & \vdots \end{bmatrix},$$

$$\hat{\gamma} = \begin{bmatrix} -\gamma_0 \\ -\gamma_1 \\ \vdots \\ -\gamma_{mn-1} \end{bmatrix}$$

and

$$\hat{X}_i = \begin{bmatrix} x_i(t+mn) \\ x_i(t+mn+1) \\ \vdots \end{bmatrix}.$$

There always exists a constant coefficient vector $\hat{\gamma}$ no matter how many rows there are in matrix X_i and vector $\hat{X}_i$. The following lemma is introduced to show that the correct coefficients in $\hat{\gamma}$ can be found by each agent within a finite amount of time steps.

Lemma 3. *For each agent in the system (1.2), $2mn$ states are sufficient to find the coefficients that guarantee the relation in (1.43)*

Proof of Lemma 3. Consider the augmented block matrix $\bar{X}_i = [X_i,\ \hat{X}_i]$. Denoting the k-th block row of $\bar{X}_i$ as $\bar{X}_i^k$, we know from (1.43) that there always exists the relation:

$$\bar{X}_i^{t+mn} = -\alpha_{mn-1} X_i^{t+mn-1} - \cdots - \alpha_0 X_i^t. \tag{1.45}$$

From the relation (1.45) we can see that starting from the $(mn+1)$-th block row of $\bar{X}_i$, any block row can be expressed by a linear combination of its preceding mn block rows. In other words, rank of the augmented matrix $\bar{X}_i$ stops increasing after the mn-th block row. Therefore, the solution $\hat{\gamma}$ to equation (1.44) where the augmented matrix has mn block rows is also a solution to (1.44) when the augmented matrix has more than mn block rows. From here, we can see that it is only necessary for an agent to store no more than $2mn$ states to obtain the coefficients so that it can update independently. ∎

Assuming the update vector $\hat{\gamma} = [\gamma_0 \ \gamma_1 \ \cdots \ \gamma_{mn-1}]$ is obtained using the method mentioned above, the update of the states of all agents can be expressed by

$$x_i(mn) = -\gamma_{mn-1}x_i(mn-1) - \cdots - \gamma_1 x_i(1) - \gamma_0 x_i(0),$$

which can be arranged as

$$x_i(mn) + \gamma_{mn-1}x_i(mn-1) + \ldots + \gamma_1 x_i(1) + \gamma_0 x_i(0) = 0. \tag{1.46}$$

Taking z-transform of (1.46), we obtain

$$\begin{gathered}(z^{mn} + \gamma_{mn-1}z^{mn-1} + \cdots + \gamma_1 z + \gamma_0)X_i(z) = \\ \sum_{j=0}^{mn-1} x_i(j) z^{mn-j} + \gamma_{mn-1}\sum_{j=0}^{mn-2} x_i(j) z^{mn-1-j} + \cdots + \gamma_1 z x_i(0).\end{gathered} \tag{1.47}$$

Since the left-hand side of (1.47) essentially contains a characteristic polynomial of W, there are exactly n poles at 1 according to Lemma 1. Therefore, the left hand side of (1.47) can be written as

$$(z-1)^n p(z) X_i(z).$$

The final value theorem for z-transforms [26] states that as long as $\lim_{t\to\infty} x(t)$ remains finite, the final value of this convergent series can be obtained by

$$\lim_{t\to\infty} x_i(t) = \lim_{z\to 1}(z-1)X_i(z). \tag{1.48}$$

Because the final value is finite and W has n poles at 1, for (1.48) to have a solution, the right-hand side of (1.47) has to contain $n-1$ zeros at 1 and can be written as

$$(z-1)^{n-1} q(z).$$

Consider $\gamma_{mn} = 1$, the final value can be expressed as

$$\lim_{k\to\infty} x_i(k) = \lim_{z\to 1}(z-1)\frac{(z-1)^{n-1}q(z)}{(z-1)^n p(z)}, \tag{1.49}$$

here $q(z) = \big[x_i(0) \ x_i(1) \ \cdots \ x_i(mn-n)\big] * \Big(\tilde{S} * \tilde{h}_2$, and $p(z) = S * \tilde{h}_1$.

The final finite-time solution step (1.37) only requires an agent's knowledge of the coefficients of characteristic equation, some of its previous states, total amount of agents in the network, and the values on the diagonal of the Pascal's triangle. All the information can be achieved without the existence of any central agent.

1.4.3 Example

Finite Time Solution of DALE

To demonstrate the capability of the finite time method, we consider the same example as shown in 1.3.3. In this case, $m = 4$ and $n = 5$. The coefficients $[\gamma_n, \cdots, \gamma_{mn}]$ are $[-0.01\ -0.09\ 0.23\ 0.52\ -1.73\ -1.04\ 6.56\ -1.58\ -12.55\ 10.37\ 9.66\ -15.93\ 2.06\ 7.53\ -5.00\ 1]$. Taking agent 1 for example, $[x(0)\ x(1)\ \cdots\ x(mn-n)]$, its states achieved using (1.4), are shown in Figure 1.5.

Columns 1 through 8

0	-0.7766	-0.7279	-0.9314	-0.9190	-1.0098	-1.0028	-1.0441
0	-0.5305	-0.1851	-0.2391	-0.1129	-0.0694	0.0034	0.0580
2.1167	0.6620	0.8158	0.5396	0.5923	0.4736	0.4925	0.4351
0	0.1138	-0.1211	-0.0035	-0.0913	-0.0598	-0.1028	-0.1006
0	0.5799	0.4972	0.6789	0.6440	0.7582	0.7618	0.8371

Columns 9 through 16

-1.0379	-1.0551	-1.0492	-1.0547	-1.0489	-1.0488	-1.0433	-1.0407
0.1114	0.1588	0.2005	0.2384	0.2711	0.3006	0.3260	0.3487
0.4425	0.4133	0.4161	0.4007	0.4014	0.3929	0.3926	0.3876
-0.1268	-0.1340	-0.1524	-0.1620	-0.1757	-0.1850	-0.1956	-0.2037
0.8557	0.9079	0.9304	0.9683	0.9901	1.0187	1.0383	1.0605

Figure 1.5.. The first 16 states of agent 1

Then according to (1.37),

$$\lim_{t\to\infty} x_i(t) = \begin{bmatrix} -0.920 \\ 0.464 \\ 0.334 \\ -0.266 \\ 1.333 \end{bmatrix}.$$

and it is indeed the solution to linear equation (1.35).

Finite Time Solution of Distributed Averaging

To demonstrate the application of the Finite-time Solution method on some distributed averaging algorithms, we consider the same problem as mentioned in Section 1.3.3.

Assuming the characteristic has been achieved to be

$$[\gamma_n, \cdots, \gamma_{mn}] = \begin{bmatrix} 1 & -2 & \frac{1}{3} & \frac{4}{3} & -\frac{5}{9} & -\frac{2}{9} & \frac{1}{9} \end{bmatrix}$$

using the aforementioned distributed method. Taking the states of all agents for example, their first 5 states are

$$[\boldsymbol{x}(0)\ \boldsymbol{x}(1)\ \boldsymbol{x}(2)\ \boldsymbol{x}(3)\ \boldsymbol{x}(4)] = \begin{bmatrix} 15.238 & 12.764 & 22.166 & 21.341 & 24.475 \\ 82.582 & 69.810 & 77.852 & 73.595 & 76.276 \\ 53.834 & 23.156 & 35.031 & 24.805 & 28.764 \\ 99.614 & 62.716 & 83.530 & 71.231 & 78.168 \\ 7.818 & 40.969 & 19.692 & 30.743 & 23.651 \\ 44.268 & 93.936 & 65.081 & 81.637 & 72.019 \end{bmatrix}.$$

$$\tilde{S} = \begin{bmatrix} 0.111 & 0.667 & -0.667 & -1.000 & 1.000 \\ 0.667 & -0.667 & -1.000 & 1.000 & 0.000 \\ -0.667 & -1.000 & 1.000 & 0.000 & 0.000 \\ -1.000 & 1.000 & 0.000 & 0.000 & 0.000 \\ 1.000 & 0.000 & 0.000 & 0.000 & 0.000 \end{bmatrix},$$

$$S = \begin{bmatrix} 0.111 & 0.667 & -0.667 & -1.000 & 1.000 \end{bmatrix},$$

$$\tilde{h}_1 = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{bmatrix},$$

$$\tilde{h}_2 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}.$$

Then, according to (1.37),

$$\lim_{t\to\infty} x_i(t) = \begin{bmatrix} 25.630 \\ 75.488 \\ 25.630 \\ 75.488 \\ 25.630 \\ 75.488 \end{bmatrix} \tag{1.50}$$

is indeed the average state.

2. COORDINATED STRING STABILITY OF TWO PLATOONS

2.1 Introduction

For the implementation of multi-agent control, automobiles, especially heavy-duty trucks, play a fundamental role in freight transportations but have tremendously large fuel consumption [27, 28]. On one hand, the capability of individual mobile vehicles have been dramatically increased by the existing adaptive cruise control (ACC), which utilizes on-board sensors such as radar and lidar to gather information of vehicles' surroundings [29]. On the other hand, techniques in cooperative adaptive cruise-control (CACC) have been demonstrated to improve a group of connected vehicles' performance by also enabling vehicles to communicate and coordinate with each other [30–32]. Information such as acceleration of a vehicle, which is usually not directly measured from other vehicles, become available under CACC [33]. Platooning, connect multiple vehicles together using CACC techniques so they drive in the same traffic lane with small distance separations, has been considered as a promising way in achieving better fuel efficiency [34], increasing traffic throughput, as well as increasing driving safety. The small distance separations can be characterized by fixed distances or certain amount of fixed headway times considering the variety of vehicles' speeds [35] and are usually small enough to lead to aerodynamic drag savings.

One of the key research problems for platoons based on CACC is the so-called string stability that measures the capability of a platoon to attenuate disturbances [36–38]. By achieving string stability, disturbances occurring at any vehicle will not be amplified through the platoon. Without string stability, vehicles driving after one vehicle under disturbance might frequently reach their accelerating or braking limits, which could ultimately lead to traffic jams or even collisions. Besides the Lyapunov stability approach [39] and the spatially invariant systems approach [40] [41] [42] in analyzing the string stability, the most popular and intuitive method that is recently attractive to researchers is the performance-oriented frequency domain approach [36]. Along this direction, previous works mainly focus on analysis of a single platoon of vehicles [36, 38, 43, 44] with mild generalization to specific types of multiple platoons of the leader-follower structure [45].

In order to fill the knowledge gap of connecting multiple platoons, this work investigates the impact of different communication structures on the string stability of two platoons, in which each platoon consists of two vehicles. Such impact is measured by the required minimum headway time for string stability under different communication topologies. An observation, which is contrast to our intuition and also the main result of this work, is that communications in a distributed way leads to smaller requirement of headway time than those in a more centralized way. This observation is validated by simulations on the frequency domain analysis of corresponding transfer functions. Communication delay is also considered in this work because it obviously plays a significant role in achieving string stability.

2.2 String Stability Problem Formulation

For vehicle platooning, given a vehicle platoon constituted by two smaller platoons in which leading platoon consists of vehicle 1 and 2 and the follower platoon consists of vehicle 3 and 4. Vehicle 1 in the leading platoon maintains a constant velocity. Each other vehicle i, $i = 2, 3, 4$, needs to maintain a predetermined constant headway time h_i from vehicle $i - 1$ driving ahead of itself. Then the desired distance between i and $i - 1$ can be expressed as

$$d_i(t) = h_i v_i(t), \tag{2.1}$$

where $v_i(t)$ is the velocity of the i-th vehicle. Here for simplicity and without losing any generality, we assume the inter-vehicle distance between vehicles at rest is 0. Suppose each vehicle i knows its own position $x_i(t)$, velocity $v_i(t)$, acceleration $a_i(t)$, and is able to measure $d_{i,i-1}(t)$, namely the distance away from vehicle $i-1$ by radar, as shown in Fig. 2.1.

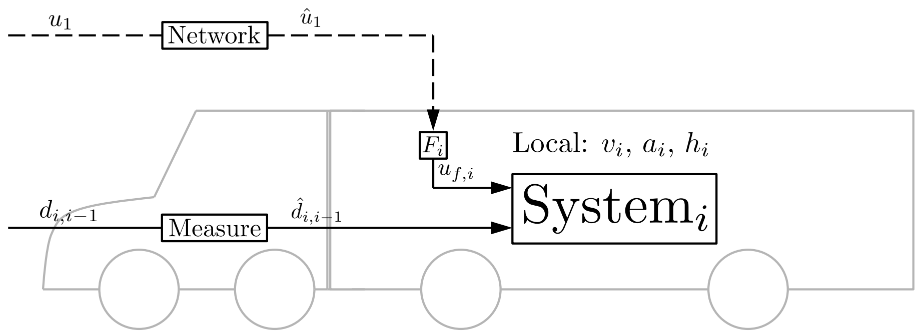


Figure 2.1.. Data transmission structure of vehicle i. a_{i-1} and $d_{i,i-1}$ are represented by $\hat{a}_{i-1}$ and $\hat{d}_{i,i-1}$, respectively, due to data transmission and measurement error.

Each vehicle i also knows the acceleration of other vehicles depending on the communication links introduced by CACC. All seven types of communication topology in terms of the frequency domain block diagrams are displayed in Fig.2.2, Fig.2.3, and Fig.2.4, in which e_i represents error between the desired value of the inter-vehicle distance and real-time measurement of it, namely, $e_i = d_i - d_{i,i-1}$, and u_i represents control input. The blue arrowed lines indicate the information flow of passing acceleration from one vehicle to others via wireless communication. Assuming ideal vehicle dynamics, we have the transfer functions $G_i(s) = s^{-2}$, $K_i(s) = \omega_{K,i}(\omega_{K,i} + s)$, $H_i(s) = 1 + h_i s$, D_i and F_i that represent the vehicle model, a PD-type control gain, and the break point, the communication delay, a feedforward filter that converts incoming information into a part of the control input respectively. Detailed derivations from time domain to frequency domain for achieving these transfer functions can be found in [35].

The overall vehicle platoon is said to be *string stable* if the oscillation in the states of a vehicle in a platoon can be attenuated along the vehicle string. Let

$$S_{X_i}(s) = \frac{X_i(s)}{X_1(s)}, i \geq 1, \tag{2.2}$$

where $X_i(s)$ is the Laplace transformation of x_i. In frequency domain with $s = j\omega$, string stability can be quantified by the magnitude of the string stability transfer function [38, 46], that is,

$$||S_{X_i}(j\omega)||_\infty \leq 1, \ i \geq 1, \tag{2.3}$$

where $||\cdot||_\infty$ is the H-infinity norm and

$$||S_{X_i}(j\omega)||_\infty = \sup_{\omega\in\mathbb{R}} ||S_{X_i}(j\omega)||.$$

A necessary and sufficient condition for string stability can be written as

$$||S_{X_i}(j\omega)||_\infty = ||\frac{X_i(j\omega)}{X_{i-1}(j\omega)}||_\infty \leq 1, \quad \text{for } i > 1. \tag{2.4}$$

Observe that

$$\frac{X_i(j\omega)}{X_{i-1}(j\omega)} = \frac{sX_i(j\omega)}{sX_{i-1}(j\omega)} = \frac{s^2X_i(j\omega)}{s^2X_{i-1}(j\omega)}.$$

Thus the string stability expressed in condition (2.4) guarantees the magnitude of oscillation in absolute position x_i, velocity $\dot{x}_i$, and acceleration $\ddot{x}_i$ do not amplify through the platoon.

Different communication topologies between the two smaller platoons will require different headway time for the overall platoon to achieve string stability. Smaller the required headway time is, more traffic throughput it allows. The **problem** of interest in this work is to identify the communication topology with the smallest required headway time from all seven configurations that will be introduced in the following.

2.3 String Stability Derivation

Using a control structure with communication delay D_i in consideration, a cascaded system can be built to represent two 2-vehicle platoons. The string stability transfer function of each communication topology will be derived in this section in order to find the minimum value of headway time h_i that ensures string stability.

2.3.1 Configuration (a)

For the well-studied preceder following strategy, the information structure can be expressed as (a) in Fig. 2.2. The string stability transfer functions can be expressed in a general form:

$$\begin{aligned} \frac{X_1}{X_0} &= \frac{G_1 K_1}{1 + H_1 G_1 K_1} \\ \frac{X_i}{X_{i-1}} &= \frac{G_i(K_i + s^2 F_i D_i)}{1 + H_i G_i K_i} \qquad i \in [2, 3, 4] \end{aligned} \tag{2.5}$$

And filter F_i of the i-th vehicle is

$$F_i = \frac{1}{H_i} \tag{2.6}$$

The filter F_i of Equation (2.6) is derived under a zero-error condition and ensures the string stable condition (2.3) when communication delay is not present ($D_i = 1$) [35].

2.3.2 Configuration (b)

Consider a condition where the leader's wireless communication range is working under ideal condition and is large enough to reach the end of the four-vehicle platoon as (b) in Fig. 2.3. The the transfer functions $S_{X_1} = \frac{X_1}{X_0}$ and $S_{X_i} = \frac{X_i}{X_{i-1}}$ can be derived as:

$$\begin{aligned} \frac{X_1}{X0} &= \frac{G_1 K_1}{1 + H_1 G_1 K_1} \\ \frac{X_2}{X_1} &= \frac{G_2(K_2 + s^2 F_2 D_2)}{1 + H_2 G_2 K_2} \\ \frac{X_3}{X_2} &= \frac{G_3(K_3 + s^2 F_3 D_3 \frac{X_1}{X_2})}{1 + H_3 G_3 K_3} \\ \frac{X_4}{X_3} &= \frac{G_4(K_4 + s^2 F_4 D_4 \frac{X_1}{X_3})}{1 + H_4 G_4 K_4} \end{aligned} \tag{2.7}$$

Equation (2.7) can be expressed in a more general form as:

$$\frac{X_i}{X_{i-1}} = \frac{G_i(K_i + s^2 F_i D_i \frac{X_1}{X_{i-1}})}{1 + H_i G_i K_i} \qquad i \in [2, 3, 4] \tag{2.8}$$

Because the acceleration of the leading vehicle is transmitted to the following vehicles in a feedforward fashion, a feedforward filter F_i is designed for each vehicle. Assuming zero inter-vehicle distance at rest, the following error of the i-th vehicle e_i is defined as

$$\begin{aligned} e_i &= d_{i,i-1} - d_i \\ &= d_{i,i-1} - h_i v_i(t), \end{aligned} \tag{2.9}$$

Where $d_{i,i-1} = x_{i-1} - x_i$ is the real time inter-vehicle distance and d_i is as defined in (2.1) with $r_i = 0$ due to zero at-rest distance assumption. Taking the following error e_i into frequency domain,

$$E_i = (\frac{X_i}{X_{i-1}})^{i-2} X_1 (1 - \frac{X_i}{X_{i-1}} H_i) \quad i \geq 2. \tag{2.10}$$

To ensure Equation (2.10) satisfies the zero-error condition, we require

$$1 - \frac{X_i}{X_{i-1}} H_i = 0 \quad i \geq 2. \tag{2.11}$$

Substituting the term $\frac{X_i}{X_{i-1}}$ with Equation (2.8), we get

$$F_i = \frac{X_{i-1}}{X_1} \frac{1}{H_i D_i G_i s^2} \quad i \geq 2. \tag{2.12}$$

Since the feedforward filter is not expected to reduce the effect of communication delay and model inaccuracy, it is assumed that there is no communication delay and the model is ideal. Therefore, we have $D_i = 1$ and $G_i = s^{-2}$. With these assumptions in mind, the feedforward filter becomes:

$$F_i = \frac{X_{i-1}}{X_1} \frac{1}{H_i} \quad i \geq 2. \tag{2.13}$$

Since, for no communication delay condition, we know $\frac{X_i}{X_{i-1}} = \frac{1}{H_i}$ from Equation (2.5). And equation (2.13) can be expanded to get

$$F_i = \begin{cases} \frac{1}{H_i}, & \text{if } i = 2 \\ \frac{\prod_{n=1}^{i-1} X_n}{\prod_{m=1}^{i-2} X_m X_1} \frac{1}{H_i}, & \text{if } i > 2. \end{cases} \tag{2.14}$$

The feedforward filter F_i can conveniently be expressed as

$$F_i = \frac{1}{H_i H_{i-1} \ldots H_2} \quad i \geq 2. \tag{2.15}$$

Substituting Equation (2.13) into (2.8),

$$\frac{X_i}{X_{i-1}} = \frac{G_i(K_i + s^2 D_i \frac{1}{H_i})}{1 + H_i G_i K_i} \quad i \geq 2 \tag{2.16}$$

Note that when there is no communication delay in presence ($D_i = 1$) and $G_i = s^{-2}$, Equation (2.16) becomes

$$\frac{X_i}{X_{i-1}} = \frac{1}{H_i} = \frac{1}{1 + h_i s} \quad i \geq 2. \tag{2.17}$$

String stability is guaranteed for (2.17) as

$$||\frac{X_i}{X_{i-1}}||_\infty = ||\frac{1}{1 + h_i s}||_\infty \leq 1 \quad i \geq 2 \tag{2.18}$$

For a platoon with uniform headway time, H_i is the same for all i and Equation (2.18) can be simplified to:

$$F_i = \frac{1}{H^{i-1}} = \frac{1}{(1 + hs)^{i-1}} \quad i \geq 2. \tag{2.19}$$

2.3.3 Configuration (c)-(g)

The communication structures (c)-(g) in Fig. 2.4 represent two 2-vehicle platoon with the communication range of the 1st and 2nd vehicle being (2,2), (3,2), (2,1), (3,1), and (1,2), respectively. The string stability transfer functions of range (2,2) are derived as:

$$\begin{aligned}
\frac{X_1}{X_0} &= \frac{G_1 K_1}{1 + H_1 G_1 K_1} \\
\frac{X_2}{X_1} &= \frac{G_2(K_2 + s^2 F_2 D_2)}{1 + H_2 G_2 K_2} \\
\frac{X_3}{X_2} &= \frac{G_3(K_3 + s^2 F_3^1 D_3^1 \frac{X_1}{X_2} + s^2 F_3^2 D_3^2)}{1 + H_3 G_3 K_3} \\
\frac{X_4}{X_3} &= \frac{G_4(K_4 + s^2 F_4^2 D_4^2 \frac{X_2}{X_3} + s^2 F_4^3 D_4^3)}{1 + H_4 G_4 K_4}
\end{aligned} \tag{2.20}$$

By deriving the equation of following error in frequency domain and setting it to zero, the filters of structure (c) are obtained as:

$$
\begin{aligned}
&F_2 = \frac{1}{H_2} \\
&F_3^2 = \frac{\alpha_3^2}{H_3} \quad F_3^1 = \frac{\alpha_3^1}{H_2 H_3} \\
&F_4^3 = \frac{\alpha_4^3}{H_4} \quad F_4^2 = \frac{\alpha_4^2}{H_3 H_4} \\
&\text{where } \alpha_i^{i-2} \geq 0,\ \alpha_i^{i-1} \geq 0,\ \alpha_i^{i-2} + \alpha_i^{i-1} = 1
\end{aligned}
\tag{2.21}
$$

Notice that F_i^k means the feedforward filter of the information that is fed forward from the k-th vehicle to the i-th vehicle. And α_i and β_i in (2.21) serve as the weights assigned to the acceleration information coming from the corresponding vehicle k. When β_is are set equal to zero, information is only transmitted from a preceder to a follower. The communication structure turns into "preceder following" and (2.20) becomes identical to (2.5). By deriving (2.6),(2.19), and (2.21), it was concluded that the frequency domain filter F_i^k has the general form of

$$
F_i^k = \frac{\alpha_i^k}{\prod_{j=k+1}^{i} H_j}, \text{ where} \sum_{\forall m \in k} \alpha_i^m = 1, 0 < k < i \tag{2.22}
$$

A filter as (2.22) can be applied to all other communication structures (d)-(g).

Remark 1. *Substituting the feedforward filters into string stability transfer functions, string stability transfer functions become a function of delay time τ_i, control gain $\omega_{k,i}$, and headway time $h_{min,i}$ and can be represented by*

$$
\frac{X_i}{X_{i-1}} = f(\tau_i, \omega_{k,i}, h_i) \tag{2.23}
$$

2.4 Simulation

In this section, we will identify the communication topology between two platoons which requires the minimum headway time for achieving string stability by simulations in MATLAB.

2.4.1 Minimum Desired Headway Time without Communication Delay

The string stability transfer function of a i-th vehicle in any platoon mentioned above with no communication delay can be expressed by (2.17). As discussed above in (2.18), the string stability of such a platoon is guaranteed for any real number h_i. Therefore, any headway time $h_i \geq 0$ can be chosen.

2.4.2 Minimum Desired Headway Time with Communication Delay

To solve for the minimum string stable headway time by putting constraint on a rational transfer function, the communication delay was approximated using Padé approximation of the form:

$$e^{-\tau s} \approx \frac{1 - \frac{\tau}{2}s + \frac{\tau^2}{10}s^2 - \frac{\tau^3}{120}s^3 \dots}{1 + \frac{\tau}{2}s + \frac{\tau^2}{10}s^2 + \frac{\tau^3}{120}s^3 \dots}. \tag{2.24}$$

For Configuration 1, a contour plot of the minimum headway time of the i-th vehicle, $h_{min,i}$, with respect to different communication delay τ_i and control gain frequency $\omega_{k,i}$ has been presented in Fig. 2.5. For all communication configurations, the minimum headway times $h_{min,i}$ of the 2-nd, 3-rd, and 4-th vehicle for the condition where communication delay τ_i is set to $100ms$, and control gain $\omega_{k,i} = 1$ are presented in Table 2.1. All communication weights α_i^k are set equal. (a)-(g) corresponds to the communication structures introduced above. The minimum headway time for the first vehicles are not presented because that belongs to the design of ACC system and is beyond the scope of this work. As shown in Table 2.1, communicating with a vehicle that is two indices away more than doubles the minimum headway time.

2.4.3 Validation

The two-2-vehicle-platoon model was tested with MATLAB simulink under both delay-free and delay-present conditions. Time history of accelerations for each vehicle is shown in Fig. 2.6. The simulated condition is the reference acceleration for the

Table 2.1.: $h_{min,i}$, minimum headway time of the i-th vehicle with $\tau = 100ms$, $\omega_{k,i} = 1$ for every vehicle.

	$h_{min,2}(s)$	$h_{min,3}(s)$	$h_{min,4}(s)$
(a)	0.387	0.387	0.387
(b)	0.387	1.001	2.415
(c)	0.387	1.000	2.002
(d)	0.387	1.000	2.010
(e)	0.387	1.000	0.387
(f)	0.387	1.000	2.002
(g)	0.387	0.387	1.000

platoon suddenly increases from $0m/s^2$ to $1m/s^2$. The result shows that string stability is achieved as the magnitude of acceleration of a following vehicle never exceeded that of its predecessor even when communication delay is present.

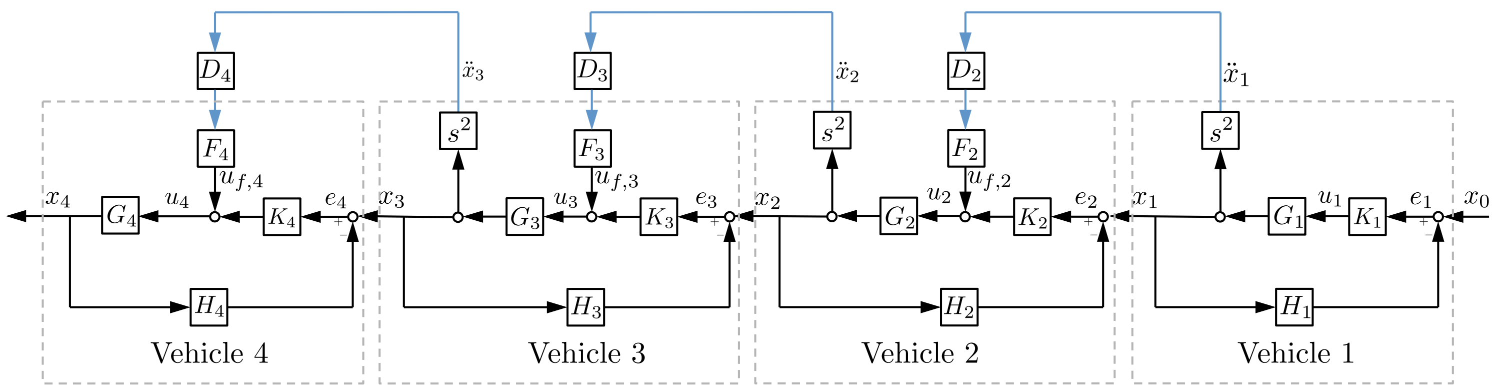


Figure 2.2.. Communication configuration (a).

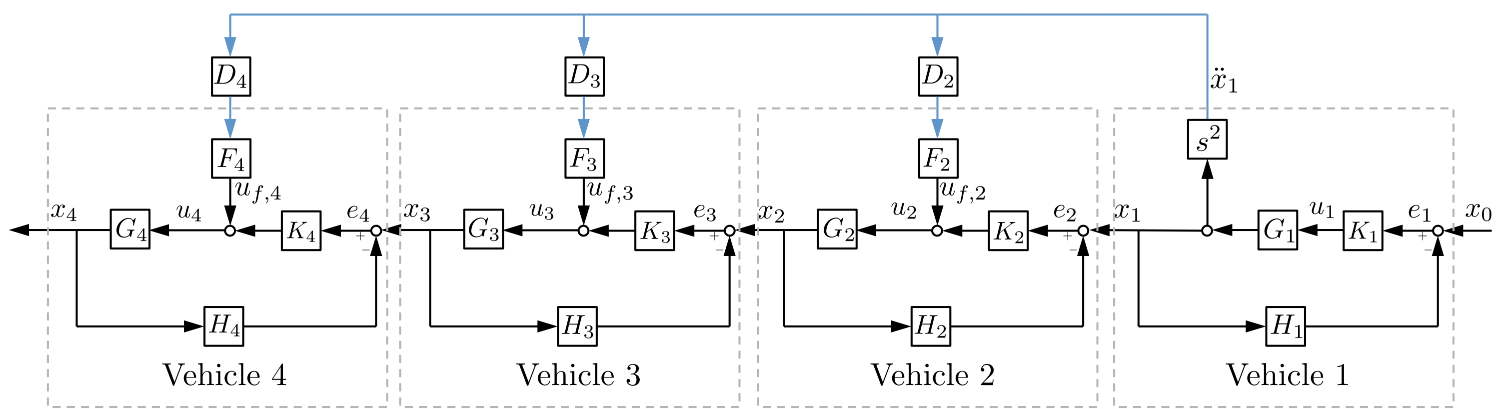


Figure 2.3.. Communication configuration (b).

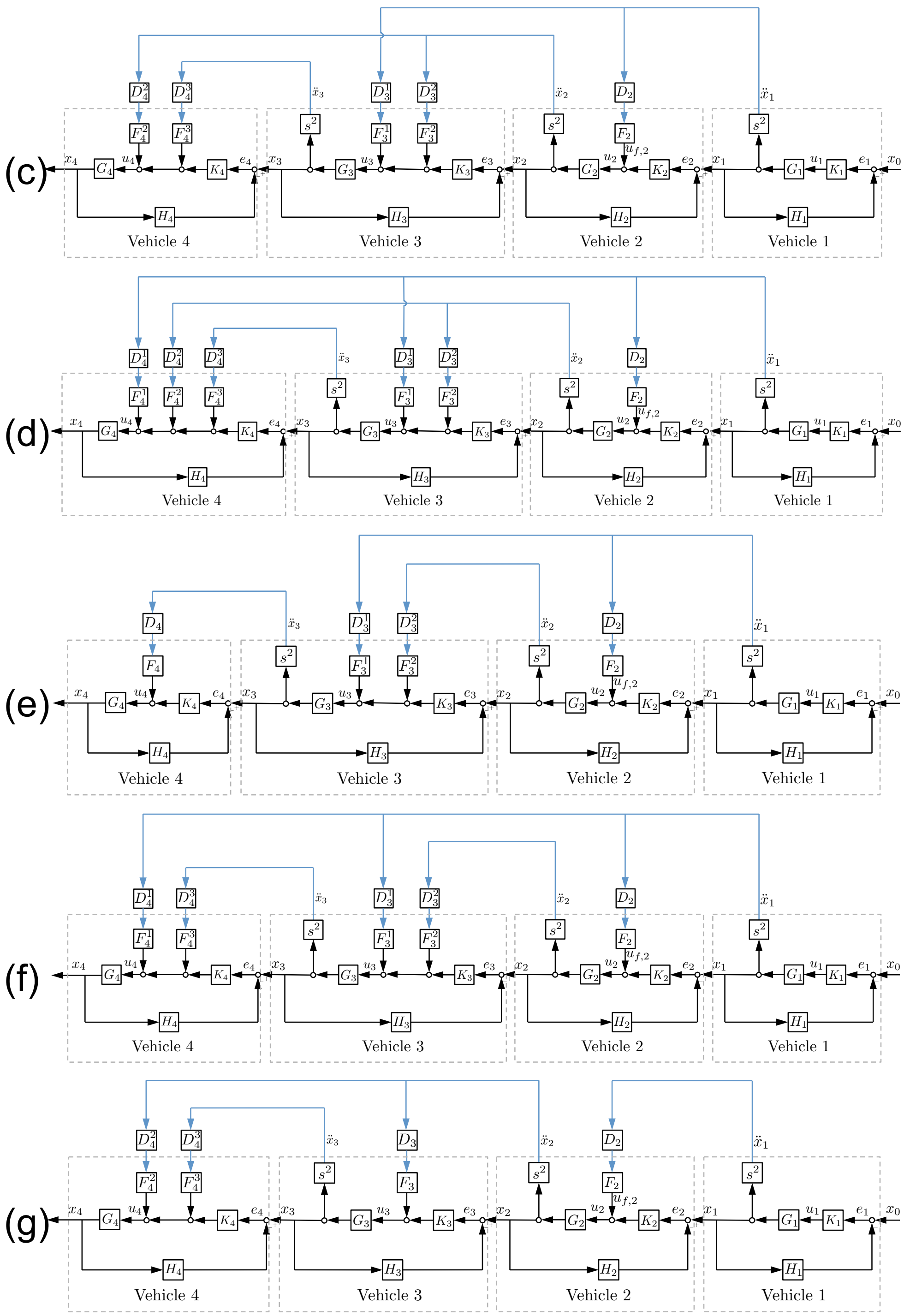


Figure 2.4.. Communication configuration (c).

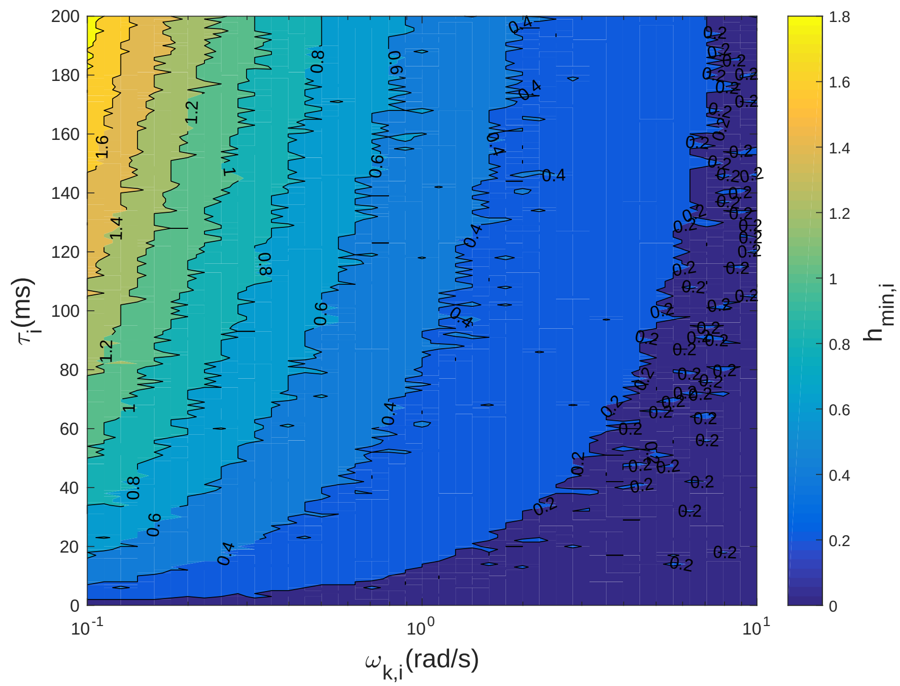
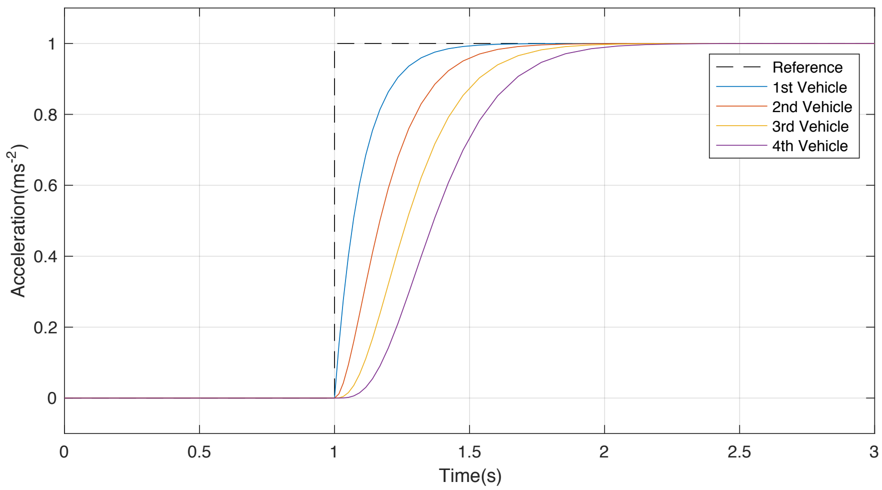
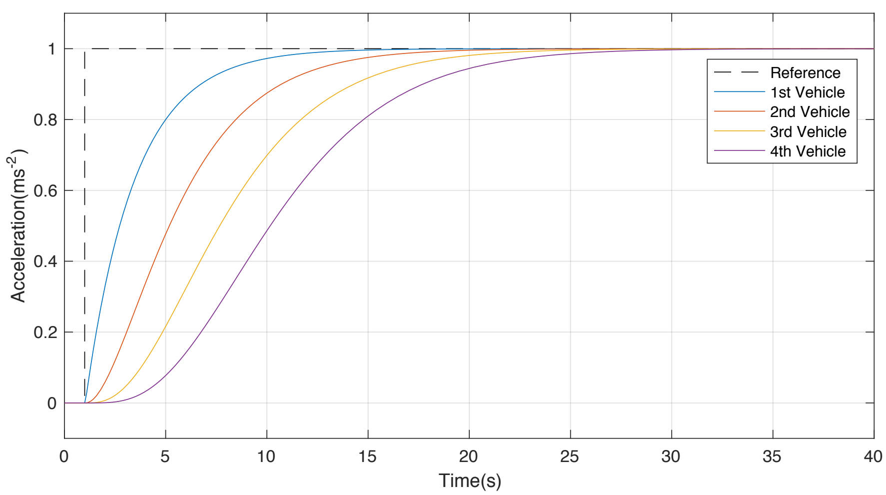


Figure 2.5.. Minimum headway time to ensure string stability for different gain frequency $\omega_{k,i}$, communication delay τ_i.



(a)



(b)

Figure 2.6.. Simulation results under both (a) delay-free and (b) delay-present conditions. $w_{k,i} = 1$ for both cases, $\tau = 0ms$, $h_i = 0.1s$ for (a) and $\tau = 100ms$, $h_i = 2.5s$ for (b).

3. CONCLUSION AND FUTURE WORK

In the first part of the work, an algorithm that accelerates some current consensus-based algorithms was introduced. The new algorithm requires each agent to store a state from the previous time step, and combine its state of previous time with current time to achieve update. The proof of faster convergence speed was done by deriving the relationship between original algorithm's system update matrix's eigenvalue and the new system update matrix eigenvalue. Some current algorithms such as Distributed Algorithm for solving Linear Equations and Distributed Averaging with symmetrical communication weight have been demonstrated to have faster convergence using the acceleration method. A method that allows reaching convergence within a finite amount of time was also introduced. The idea of finite-time solution was a result of the combination of characteristic polynomial and discrete-time final value theorem. The application of finite-time solution was also demonstrated on the two original algorithms as mentioned above as examples. For future work, the Accelerated Consensus-based Algorithm should be extended to the case where the original system update matrix has complex eigenvalues. This way, the application of the algorithm can be extended to more general consensus-based algorithms. The finite-time method should be modified so that it requires each agent to have less storage space, and so that it requires less computing power from each agent.

In the second part, the string stability of different communication structures have been derived and a general method for finding a feedforward filter was developed. It is shown numerically by configuration (b) of Table 2.1 that $h_{min,i}$ becomes larger when the i-th vehicle receives information from a vehicle that is located farther forward in the platoon in order to ensure string stability. In other words, when all conditions the same, the vehicle that receives information from a leader that is farther away index-wise has to have longer headway time to ensure string stability. However, for real-world systems, the headway time difference does not have to be as large as shown in this paper because real systems do not have infinite frequencies while the numerical results of this paper are obtained under a frequency range that goes to infinity. For future work, more realistic background should be considered. For example, each vehicle in the platoon should have different parameters, and the string stability condition should be relaxed so that it does not necessarily cover infinite frequency. This way, a more feasible headway time can be derived. Further, the comparison between headway times should be given by mathematical derivation instead of simulation.

REFERENCES

REFERENCES

[1] G. Weiss. Multiagent systems: A modern approach to distributed artificial intelligence. *The MIT Press*, 1999.

[2] J.N. Tsitsiklis, D.P. Bertsekas, and M. Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE Transactions on Automatic Control*, 31(9):803–812, 1986.

[3] R. Olfati-Saber and R. M. Murray. Consensus protocols for networks of dynamic agents. *Proceedings of the American Control Conference*, pages 951–956, 2003.

[4] A. Jadbabaie, J. Liu, and A.S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, 2003.

[5] L. Xiao and S. Boyd. Fast linear iterations for distributed averaging. *Systems and Control Letters*, 53:65–78, 2004.

[6] J. Lu and Y. Tang. Distributed asynchronous algorithms for solving positive definite linear equations over networks-part i: Agent networks. *Proceedings of First IFAC Workshop on Estimation and Control of Network Systems*, pages 22–26, 2004.

[7] L. Moreau. Stability of multi-agent systems with time-dependent communication links. *IEEE Transactions on Automatic Control*, 50(2):169–182, 2005.

[8] W. Ren and R.W. Beard. Consensus seeking in multiagent systems under dynamically changing interaction topologies. *IEEE Transactions on Automatic Control*, 50(5):655–661, 2005.

[9] S. Sundaram and C. N. Hadjicostis. Distributed function calculation and consensus using linear iterative strategies. *IEEE Journal on Selected Areas in Communications*, pages 650–660, 2008.

[10] A. Nedic and A. Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, pages 48–61, 2009.

[11] A. Y. Kibangou. Finite-time average consensus based protocol for distributed estimation over awgn channels. *IEEE Conference on Decision and Control and European Control Conference*, pages 5595–5600, 2011.

[12] S. Mou, J. Liu, and A.S. Morse. A distributed algorithm for solving a linear algebraic equation. *IEEE Transactions on Automatic Control*, pages 2863–2878, 2015.

[13] T. Charalambous, Y. Yuan, T. Yang, W. Pan, C. N. Hadjicostis, and M. Johansson. Distributed finite-time average consensus in digraphs in the presence of time delays. *IEEE Transactions on Control of Network Systems*, pages 370–381, 2015.

[14] X. Wang, S. Mou, and D. Sun. Improvement of a distributed algorithm for solving linear equations. *Transactions on Industrial Electronics*, 64:3113–3117, 2017.

[15] S. Muthukrishnan, B. Ghosh, and Schultz M. H. First- and second-order diffusive methods for rapid, coarse, distributed load balancing. *Theory of Computing Systems*, 31:331–354, 1998.

[16] T. C. Aysal, B. N. Oreshkin, and M. J. Coates. Accelerated distributed average consensus via localized node state prediction. *IEEE Transactions on Signal Processing*, 57:1563–1575, 2009.

[17] J. Liu, B.D.O. Anderson, M. Cao, and A.S. Morse. Analysis of accelerated gossip algorithms. *Automatica*, pages 873–883, 2013.

[18] S. Sundaram and C. N. Hadjicostis. Finite-time distributed consensus in graphs with time-invariant topologies. *American Control Conference*, pages 711–716, 2007.

[19] T. M. D. Tran and A. Y. Kibangou. Distributed design of finite-time average consensus protocols. *4th IFAC Workshop on Distributed Estimation and Control in Networked Systems*, pages 227–233, 2013.

[20] J. M. Hendrickx, R. M. Jungers, A. Olshevsky, and G. Vankeerberghen. Graph diameter, eigenvalues and minimum-time consensus. *Automatica*, pages 635–640, 2013.

[21] S. Mou and A. S. Morse. Finite-time distributed averaging. *American Control Conference*, pages 5260–5263, 2014.

[22] G. Oliva, R. Setola, and C. N. Hadjicostis. Distributed finite-time average-consensus with limited computational and storage capability. *IEEE Transactions on Control of Network Systems*, pages 380–391, 2016.

[23] D. Young. Iterative methods for solving partial difference equations of elliptic type. *Transactions of the American Mathematical Society*, 76:92–111, 1954.

[24] G. H. Golub and R. S. Varga. Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second order richardson iterative methods. *Numerische Mathematik*, 3:147–156, 1961.

[25] S. Mou and A. S. Morse. A fixed-neighbor, distributed algorithm for solving a linear algebraic equation. *European Control Conference*, pages 2269–2273, 2013.

[26] A. D. Poularikas. *The transforms and applications handbook, second edition*. CRC Press, Boca Raton, 2010.

[27] 2012 commodity flow survey: United states, final release. Technical report, U.S. Department of Transportation and Bureau of Transportation Statistics and U.S. Department of Commerce and Census Bureau, Washington, DC, December 2014.

[28] Highway statistics 2012. Technical report, U.S. Department of Transportation Federal Highway Administration, Washington, DC, 2014.

[29] M.R.I. Nieuwenhuijze. String stability analysis of bidirectional adaptive cruise control. *TU/e Master's Thesis*, 2010.

[30] A. Geiger, M. Lauer, F. Moosmann, B. Ranft, H. Rapp, C. Stiller, and J. Ziegler. Team annieway's entry to the 2011 grand cooperative driving challenge. *IEEE Transactions on Intelligent Transportation Systems*, pages 1008–1017, September 2012.

[31] R. Kianfar, B. Augusto, A. Ebadighajari, U. Hakeem, J. Nilsson, A. Raza, R. Tabar, N.V. Irukulapati, C. Englund, P. Falcone, S. Papanastasiou, L. Svensson, and H. Wymeersch. Design and experimental validation of a cooperative driving system in the grand cooperative driving challenge. *IEEE Transactions on Intelligent Transportation Systems*, pages 994–1007, September 2012.

[32] L. Guvenc, I.M.C. Uygan, K. Kahraman, R. Karaahmetoglu, I. Altay, M. Senturk, M.T. Emirler, A.E.H. Karci, B.A. Guvenc, E. Altug, M.C. Turan, O.S. Tas, E. Bozkurt, U. Ozguner, K. Redmill, A. Kurt, and B. Efendioglu. Cooperative adaptive cruise control implementation of team mekar at the grand cooperative driving challenge. *IEEE Transactions on Intelligent Transportation Systems*, pages 1062–1074, September 2012.

[33] K.C. Dey, L. Yan, X. Wang, Y. Wang, H. Shen, M. Chowdhury, L. Yu, C. Qiu, and V. Soundararaj. A review of communication, driver characteristics, and controls aspects of cooperative adaptive cruise control(cacc). *IEEE Transactions on Intelligent Transportation Systems*, pages 491–509, February 2016.

[34] M. P. Lammert, A. Duran, J. Diez, K. Burton, and A. Nocholson. Effect of platooning on fuel consumption of class 8 vehicles over a range of speeds, following distances, and mass. *SAE 2014 Commercial Vehicle Engineering Congress*, September 2014.

[35] R.P.A. Vugts. String-stable cacc design and experimental validation. *TU/e Master's Thesis*, 2009.

[36] G. J. L. Naus, R. P. A. Vugts, J. Ploeg, M. J. G. van de Molengraft, and M. Steinbuch. String-stable cacc design and experimental validation: A frequency-domain approach. *IEEE Transactions on Vehicular Technology*, pages 4268–4279, November 2010.

[37] M.R.I. Nieuwenhuijze, T. van Keulen, S. Oncu, B. Bonsen, and H. Nijmeijer. Cooperative driving with a heavy-duty truck in mixed traffic: Experimental results. *IEEE Transactions on Intelligent Transportation Systems*, pages 1026–1032, September 2012.

[38] S. Oncu, J. Ploeg, N. van de Wouw, and H. Nijmeijer. Cooperative adaptive cruise control: Network-aware analysis of string stability. *IEEE Transactions on Intelligent Transportation Systems*, pages 1527–1537, August 2014.

[39] D. Swaroop and J.K. Hedrick. String stability of interconnected systems. *IEEE Transactions on Automatic Control*, pages 1806–1810, March 1996.

[40] R. D'Andrea and G.E. Dullerud. Distributed control design for spatially interconnected systems. *IEEE Transactions of Automatic Control*, pages 1478–1495, 2003.

[41] B. Bamieh, F. Paganini, and M.A. Dahleh. Distributed control of spatially invariant systems. *IEEE Transactions of Automatic Control*, pages 1091–1107, July 2002.

[42] M. Jovanovic and B. Bamieh. Lyapunov-based distributed control of systems on lattices. *IEEE Transactions of Automatic Control*, pages 422–433, 2005.

[43] T. Stanger and L. del Re. A model predictive cooperative adaptive cruise control approach. *American Control COnference*, pages 1374–1379, June 2013.

[44] X.Y. Lu, J.K. Hedrick, and M. Drew. Acc/cacc - control design, stability and robust performance. *Proceedings of the American Control Conference*, pages 4327–4332, 2002.

[45] A.A. Peters, R.H. Middleton, and O. Mason. Leader tracking in homogeneous vehicle platoons with broadcast delays. *Automatica*, pages 64–74, 2014.

[46] J. Ploeg, N. van de Wouw, and H. Nijmeijer. Lp string stability of cascaded systems: Application to vehicle platooning. *IEEE Transactions on Control Systems Technology*, pages 786–793, March 2014.