Absolute Stability of Weakly Non-Linear Network Systems

Sharmin Akthar and Matthew C Turner

***Abstract*— This paper presents a Linear matrix inequality (LMI) approach for the absolute stability analysis of multi-agent Systems with Lur'e dynamics. Using mathematical manipulation and common LMI tools, an LMI incorporates the network property (via its Laplacian matrix) and yields sufficient conditions for absolute stability of a Lure-type multi-agent system. Using a state-similarity transformation based on an eigenvalue decomposition of the Laplacian matrix, a new scalable algorithm for the design of state-feedback controllers of these (strongly) connected communication networks is proposed. Remarkably, the algorithm is independent of the size of the network N.**

I. INTRODUCTION

Control of distributed networks of agents is one of most important topics in contemporary systems and control theory. The applications of Multi-agent systems (MASs) are vast, with a wide variety of practical systems such as flocking in autonomous vehicle swarms, coordination of satellites, etc. [4]. Unfortunately, classical controller design techniques are *inherently unsuitable* for achieving stability in networks as the networks are often multi-variable in nature and contain various non-standard constraints associated with the network structure. Therefore, over the past decade or so, theories have been developed to cope with the new challenges presented by such systems.

A distributed system is a group of network components that work together to act as a single entity. For them to act as a single entity they must agree on 'some' data value i.e. achieve consensus. To obtain this, there are algorithms which instruct each agent, also known as consensus protocols. These protocols must be fault tolerant, resilient, and designed to deal with a limited number of faulty processes. The majority of these algorithms make the strong assumption that the plant is *linear*, whereas in reality, all practical control systems are *nonlinear* to some degree.

A class of nonlinear systems, Lur'e systems, are representative models of many physical systems, such as Chua circuits, regulatory networks, etc. [5]. In recent years, Lur'e dynamics have been increasingly investigated, with the first Lur'e MAS researched in [7]. This paper will investigate these nonlinear dynamics by exploring the properties of a static nonlinearity $\phi(\cdot)$ and the amount of knowledge required of this function, in order for networks of this type of system to reach a consensus.

The general approach for studying stability of nonlinear systems was proposed by mathematician Alexandr M. Lyapunov and has since, been extensively researched. [2] The central idea of Lyapunov's 2nd method is to find a so-called Lyapunov function satisfying certain properties, which then can be used to prove stability of the nonlinear system. The approach is popular since it enables one to draw conclusions about the system's stability *without* needing to solve the underlying differential equations.

Recent research has shown how a search for a certain class of Lyapunov function, the *quadratic Lyapunov function* can be cast as a Linear matrix inequality (LMI) feasibility problem. In this paper, a similar approach is followed, and by making a certain choice of Lyapunov function, we formulate a matrix inequality which can guarantee the stability of the networked system. This results in a large LMI, which is then manipulated into a suitable and more compact LMI problem (LMIP) format via change of variables, congruence transformation and the S-procedure.

Graph theory has been used to represent the communication network, with the Laplacian matrix and its spectral decomposition being crucial to the reduction of the large LMIP. This has also allowed us to propose a new scalable algorithm, independent to the size of the network, guaranteeing absolute stability of the network. This tests the two most extreme eigenvalues of the Laplacian matrix, which is beneficial due to only two LMIPs needing to be computed, therefore suitable for MAS of large scales and their applications. Additionally, the Lur'e MAS is then simulated for a given network and state space system, demonstrating consensus of each state component for all agents, $x_{i1,i2,i3}$. The The work here provides further room for future work and a proof of consensus in Lur'e MAS.

The paper is structured as follows. The next section collect various preliminary technical results; Section III formulates the absolute stability problem for networked control systems and provides a scalable solution; following this a Matlab simulation example is provided; the paper ends with some conclusions.

A. Notation

Matrix Q is defined to be positive definite ($Q > 0$) and positive semi-definite ($Q \geq 0$) if $x^T Q x > 0 \quad \forall x \neq 0$ and $x^T Q x \geq 0 \quad \forall x$ respectively. Q must be a square, symmetric matrix, with all eigenvalues being positive real, or positive non-zero respectively. Matrix $P = -Q$ is said to be negative (semi-)definite if Q is positive (semi-)definite.

$$\therefore \text{ for } Q > 0 \Rightarrow P < 0 \text{ and } Q \geq 0 \Rightarrow P \leq 0$$

A common alternative is $Q \succeq 0, Q \succ 0, Q \preceq 0, Q \prec 0$

for positive semi-definite and positive-definite, negative semi-definite and negative-definite matrices, respectively.

The *Kronecker product* $A \otimes B$ of the matrices $A = \begin{bmatrix} a_{ij} \end{bmatrix}$ $(m \times n)$ and $B = \begin{bmatrix} b_{ij} \end{bmatrix}$ $(p \times r)$ is defined by:

$$A \otimes B := \begin{bmatrix} a_{11}B & a_{12}B & \dots & a_{1n}B \\ a_{21}B & a_{22}B & \dots & a_{2n}B \\ \vdots & \vdots & \dots & \vdots \\ a_{m1}B & a_{12}B & \dots & a_{mn}B \end{bmatrix}$$

and it is a $(mp \times nr)$ matrix. The dynamics of a network of identical agents is much easier to describe using Kronecker products. *Note: A and B do not necessarily have compatible dimensions.*

II. PRELIMINARIES

A. Lyapunov Stability

Lyapunov's Theorem gives *sufficient* conditions for (asymptotic) stability of nonlinear systems. Consider a non-linear autonomous system

$$\dot{x} = f(x) \quad f(\cdot) : \mathbb{R}^n \mapsto \mathbb{R}^n$$

The origin $x = 0$ is *globally (asymptotically) stable* if there exists a continuously differentiable, positive definite function $V(x)$ with negative (semi-) definite derivative.

1) Quadratic Lyapunov Functions: Lyapunov's method requires one to choose a positive definite Lyapunov function and then prove that its derivative is negative semi/definite. This is a difficult task in general, but in the case of linear systems, it is both necessary and sufficient to confine one's attention to linear systems.

Considering linear system

$$\dot{x} = f(x) = \underbrace{Ax}_{linear} \quad \Re(\lambda_i(A)) < 0 \quad \forall i \tag{1}$$

If A has all negative eigenvalues, there exists a $P > 0$ satisfying the Lyapunov equation:

$$A^T P + PA = -Q < 0 \tag{2}$$

It is easy to see that with $V(x) = x'P(x)$,

$$\dot{V}(x) = \frac{\partial V(x)}{\partial x}\dot{x} \tag{3}$$
$$= x^T(A^T P + PA)x \tag{4}$$

Thus for $\dot{V}(x) < 0$ it follows that equation (2) must hold. Although quadratic Lyapunov functions are a small subset of the available choice for Lyapunov functions, they have found favour amongst control engineers since i) many systems can be modelled as a linear system plus a perturbation; and ii) searches over quadratic Lyapunov functions can often be cast as LMI problems which are computationally tractable.

B. Linear Matrix Inequalities

Linear matrix inequalities (LMIs) are matrix inequalities which are linear or affine in a set of matrix variables. Linear matrix inequality problems are a sub-class of semi-definite programming problems with the key advantage that they are both computuationally tractable, with various LMI solvers available. Their significant to control theorists is that many linear and some nonlinear stability and performance problems can be case as LMI's.

The basic structure of an LMI is:

$$F(x) = F_0 + \sum_{i=1}^{m} x_i F_i > 0 \tag{5}$$

where $x \in \mathbb{R}^m$ is a variable, F_0 and F_1 are constant symmetric real matrices that are given. The basic LMI problem (LMIP), the Feasibility Problem, is that given an LMI $F(x) > 0$, to find a feasible x, x_{feas} such that $F(x_{feas}) > 0$. The more common format of an LMI is that where the the variable x is composed of one or many matrices, whose columns have been stacked as a vector:

$$F(x) = F(X_1, X_2, ..., X_n) \tag{6}$$

where $X_i \in \mathbb{R}^{q_i \times p_i}$ is a matrix, $\sum_{i=1}^{n} q_i \times p_i = m$, and the columns of all matrix variables are stacked up to form a single vector variable.

Typically, such an $F(x)$ would have the structure

$$F(x) = F(X_1, X_2, ..., X_n)$$
$$= F_0 + G_1 X_1 H_1 + G_2 X_2 H_2 + ... \tag{7}$$
$$= F_0 + \sum_{i=1}^{n} G_i X_i H_i > 0 \tag{8}$$

where F_0, G_i, H_i are given matrices, X_i are the matrix variables sought.

Often problems arising in control engineering do not initially take the form of an LMI problem and so various tools are used to re-cast them as LMI problems. An exhaustive discussion of these techniques is beyond the scope of the report, but a number are discussed below.

1) The S-procedure: Enables one to combine several quadratic inequalities into one single inequality (generally with some conservatism). In control engineering there are cases where we would like to ensure a single quadratic function of $x \in \mathbb{R}^m$ is such that:

$$F_0(x) \leq 0; \quad F_0(x) \triangleq x^T A_0 x + 2b_0 x + c_0 \tag{9}$$

whenever certain other quadratic functions are positive semi-definite, i.e. when

$$F_i(x) \geq 0; \quad F_i(x) \triangleq x^T A_i x + 2b_0 x + c_0, \quad i \in \{1, 2, ..., n\} \tag{10}$$

To illustrate the S-procedure, consider $i = 1$, for simplicity. We would like to ensure $F_0(x) \leq 0, \forall x$ such that $F_1(x) \geq 0$. Now, if there exists a positive (or zero) scalar, τ, such that

$$F_{aug}(x) \triangleq F_0(x) + \tau F_1(x) \leq 0 \quad \forall x \quad \text{such that} \quad F_1(x) \geq 0 \tag{11}$$

$F_{aug}(x) < 0$ implies $F_0(x) \leq 0$ if $\tau F_1(x) \geq 0$ because $F_0(x) \leq F_{aug}(x)$ if $F_1(x) \geq 0$. Extending this idea to q inequality constraints obtains:

$$F_0(x) \leq 0 \quad \text{whenever} \quad F_i(x) \geq 0 \tag{12}$$

holds if

$$F_0(x) + \sum_{i=1}^{q} \tau_i F_i(x) \leq 0, \quad \tau_i \geq 0 \tag{13}$$

In general the S-procedure is conservative, the inequality above implies inequality prior to that, but not vice versa. The usefulness of the S-procedure is in the possibility of including τ_i as variables in an LMI problem.

In summary, the S-Procedure (or S-Lemma) is the method of combining quadratic constraints to yield an LMI.

Example: Finding a matrix variable $P > 0$ such that

$$\begin{bmatrix} x \\ z \end{bmatrix}^T \begin{bmatrix} A^T P + PA & PB \\ B^T P & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} < 0$$

whenever $x \neq 0$ and z satisfies the constraint

$$z^T z \leq x^T C^T C x$$

which is equivalent to

$$x^T C^T C x - z^T z \geq 0$$

or

$$\begin{bmatrix} x \\ z \end{bmatrix}^T \begin{bmatrix} C^T C & 0 \\ 0 & -I \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} \geq 0$$

The two quadratic constraints can thus be combined with the S-procedure to yield the LMI:

$$\begin{bmatrix} A^T P + PA + \tau C^T C & PB \\ B^T P & -\tau I \end{bmatrix} < 0$$

in the variables $P > 0$ and $\tau \geq 0$

2) Congruence Transformation: This method uses a change of variable and some matrix properties to transform Bilinear Matrix Inequalities (BMIs) to Linear Matrix Inequalities (LMIs). This method preserves the definiteness of the matrices that undergo congruence transformation.

For a given positive definite matrix $Q \in \mathbb{R}^{n \times n}$, for any real matrix $W \in \mathbb{R}^{n \times n}$ such that $\text{rank}(W) = n$, the matrix inequality

$$Q < 0 \tag{14}$$

is satisfied if and only if the following inequality holds

$$WQW^T < 0 \tag{15}$$

Example: Consider the matrix variables $P > 0, V > 0$ and F, with the remaining matrices as constants. The matrix inequality given by

$$Q = \begin{bmatrix} A^T P + PA & PBF + C^T V \\ F^T B^T P^T + V^T C & -2V \end{bmatrix} < 0$$

is linear in variable V and bilinear in the variable pair (P, F). Choose the matrix $\text{diag}(P^{-1}, V^{-1})$ To obtain the equivalent BMI given by

$$WQW^T = \begin{bmatrix} P^{-1} A^T + AP^{-1} & BFV^{-1} + P^{-1} C^T \\ V^{-T} F^T B^T + CP^{-T} & -2V^{-1} \end{bmatrix} < 0$$

Using a change of variable $X = P^{-1}, U = V^{-1}$ and $L = FV^{-1}$ the BMI above becomes

$$WQW^T = \begin{bmatrix} XA^T + AX & BL + XC^T \\ L^T B^T + CX & -2U \end{bmatrix} < 0$$

which is an LMI of variables X, U and F. Note that the original variables can be recovered by inverting X and U.

C. Lure Systems

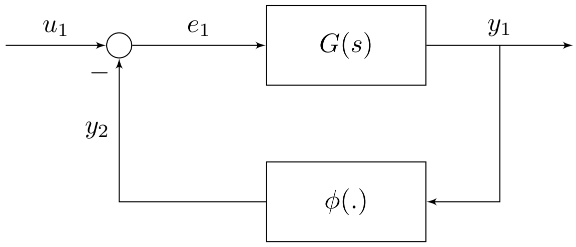

Fig. 1. Block diagram of system with static nonlinearity

An important class of nonlinear system are so called *Lur'e systems* which consist of a linear element and a static nonlinear element. Such systems are useful for modelling actuator and sensor nonlinearities in otherwise linear systems. A typical Lur'e system is shown in Figure 1, where $G(s)$ represents the linear part of the system and $\phi(\cdot)$ represents the static nonlinear element. It can be seen that Lur'e systems are a type of *weakly nonlinear system*

Static non-linearities have no states and are described by their input-output maps. The idea behind *absolute stability analysis* is to guarantee that the Lur'e system is stable for a class of nonlinearities (to be described shortly) thereby meaning that the system will *absolutely* be stable for any nonlinearity in that class.

1) Sector boundedness: An important class of static nonlinearity is that of *sector bounded* nonlinearities. A single sector bounded nonlinearity is depicted in Figure 2. The static nonlinearity in this figure, $\phi(.) : \mathbb{R} \mapsto \mathbb{R}$, with the Sector $[0, \alpha]$ means the upper boundary is α-gradient line, and lower boundary is the zero gradient line. This sector condition satisfies the inequality:

$$0 \leq \frac{\phi(x)}{x} \leq \alpha \tag{16}$$

Implying:

$$\alpha - \frac{\phi(x)}{x} \geq 0 \qquad \frac{\phi(x)}{x} \geq 0 \tag{17}$$

$$\Rightarrow \frac{\phi(x)}{x}(\alpha - \frac{\phi(x)}{x}) \geq 0 \tag{18}$$

$$= \frac{\phi(x)}{x}\alpha - \frac{\phi(x)^2}{x^2} \geq 0 \tag{19}$$

$$\Rightarrow x^2(\frac{\phi(x)}{x}\alpha - \frac{\phi(x)^2}{x^2}) \geq 0 \tag{20}$$

$$\Rightarrow \alpha\phi(x)x - \phi(x)^2 \geq 0 \tag{21}$$

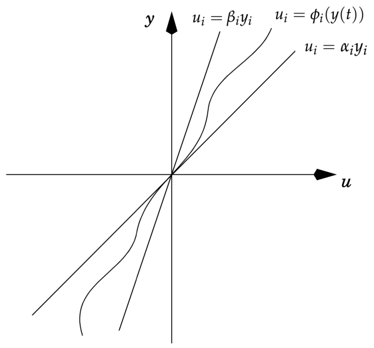

Fig. 2. Visualisation of sector bounded nonlinearity

Therefore, the static nonlinearity $\phi(.)$ bounded by Sector $[0, \alpha]$ can be represented by the inequality $\alpha\phi(x)x - \phi(x)^2 \geq 0$ which can then in turn be written as a matrix inequality.

In the vector case, that is when $\phi(.) : \mathbb{R}^m \mapsto \mathbb{R}^m$, then the notion of sector boundedness is similar to the scalar case, but in this case, $\phi(\cdot) \in \text{Sector}[0, \mathcal{A}]$ is taken to mean that nonlinearity $\phi(\cdot)$ satisfes the following condition

$$\phi(x)'U(\mathcal{A}x - \phi(x)) \geq 0 \quad \forall x \in \mathbb{R}^m \tag{22}$$

and all diagonal matrices $U > 0$, where $\mathcal{A} \in \mathbb{R}^{mxm}$ is some positive definite (symmetric) matrix.

2) Loop Transformation: The standard form of the Lur'e system is achieved by applying a loop transformation to the general Lur'e system. The general Lur'e system has the static nonlinearity bounded by sector $[\alpha_i, \beta_i]$, the general sector condition is $\alpha_i\sigma^2 \leq \sigma\phi_i(\sigma) \leq \beta_i\sigma^2 \quad \forall\sigma \in \mathbb{R}$. In summary, the loop transformation maps nonlinearities in sector $[\alpha_i, \beta_i]$ into a standard sector, for example, sector $[0, 1]$.

Remarks:

- A related concept is that of *passivity*. In this case, it is assumed that

$$\phi(x)'x \geq 0$$

which is equivalent to $\phi(\cdot)$ being a **first-third quadrant** nonlinearity. This creates a strong requirement on the linear element of the systems for stability to be ensured.

- It is straightforward to see that if one has a decentralised nonlinearity

$$\phi(\cdot) = \begin{bmatrix} \phi_1(.\cdot) \\ \vdots \\ \phi_m(\cdot) \end{bmatrix}$$

where $\phi_i(\cdot) \in \text{Sector}[0, \alpha_i], \quad \forall i$, then the nonlinearity $\phi(\cdot)$ is such that

$$\phi(\cdot) \in \text{Sector}[0, \mathcal{A}]$$

where $\mathcal{A} = \text{diag}(\alpha_1, \ldots, \alpha_m)$. This property will be used later in the paper.

D. Absolute stability of Lur'e systems

Consider the system in Figure 1. Assume that $G(s)$ has state space realisation

$$\dot{x} = Ax + Be_1 \tag{23}$$
$$y_1 = Cx \tag{24}$$

and that the sector nonlinearity $e_1 = \phi(y_1)$[1] is such that

$$\phi \in \text{Sector}[0, \mathcal{A}]$$

It will now be shown that global asymptotic stability of this class of systems can be obtained by invoking the Lyapunov arguments of Section II-A and framing the problem as an LMI as described in Section II-B.

Choosing a quadratic Lyapunov function, $V(x) = x'Px$, it follows that

$$\dot{V}(x) = x'(A'P + PA)x + 2x'PB\phi(y_1) \tag{25}$$

It is difficult to conclude that such an inequality is negative definite, so the sector property (22) is recalled and the S-procedure is used to adjoin this to equation (25):-

$$\begin{aligned} \dot{V}(x) &\leq x'(A'P + PA)x + 2x'PB\phi(y_1) \\ &\quad + 2\phi(y_1)'W(\mathcal{A}Cx - \phi(y_1)) \qquad (26) \\ &= \begin{bmatrix} x \\ \phi(y_1) \end{bmatrix}' \begin{bmatrix} A'P + PA & PB + C'W \\ \star & -2W \end{bmatrix} \begin{bmatrix} x \\ \phi(y_1) \end{bmatrix} \qquad (27) \end{aligned}$$

where $W > 0$ is a diagonal matrix. Thus, global asymptotic stability of the system in Figure 21 is ensured if the following matrix inequality is satisfied:

$$\begin{bmatrix} A'P + PA & PB + C'W \\ \star & -2W \end{bmatrix} < 0 \quad P, W > 0$$
$$(W \quad \text{diagonal}) \tag{28}$$

This matrix inequality is linear in the matrix variables P and W and hence stability can be established efficiently. Note that this LMI ensures that the Lur'e system in Figure 1 is stable for *any* nonlinearity in $\text{Sector}[0, \mathcal{A}]$ i.e the system is absolutely stable.

III. MAIN CONTRIBUTION

A. Problem Formulation

A connected undirected graph network with unweighted edges is used to represent a Multi-Agent System (MAS). The network of agents have non-linear dynamics, similar to the Lur'e system discussed in the previous section:

$$\dot{x}_i(t) = Ax_i(t) + Be_i(t) \qquad i \in \{1, ..., N\} \tag{29}$$

where $x_i \in \mathbb{R}^n$ is the state of the i'th agent, $e_i \in \mathbb{R}$ is i'th agent input, and where the control is applied through the scalar sector-bounded nonlinearity $\phi_i(.) : \mathbb{R} \mapsto \mathbb{R}$

$$e_i = \phi_i(u_i(t)), \quad i = \{1, ..., N\} \tag{30}$$

[1]Since, global asymptotic stability is of interest, $u_1 \equiv 0$ in this analysis; it actually follows that the results also guarantee external stability under modest conditions, so no loss of generality is implied

By loopshifting arguments introduced earlier, is is assumed, without loss of generality, that

$$\phi_i \in \text{Sector}[\alpha_i, \beta_i] \rightarrow [0, 1] \tag{31}$$

The associated sector inequality

$$\phi_i[u_i - \phi_i] \geq 0 \tag{32}$$
$$w_i\phi_i[u_i - \phi_i] \geq 0, w_i \geq 0 \tag{33}$$

Summing the N sector inequalities then gives

$$\sum_{i=1}^{N} w_i\phi_i[u_i - \phi_i] \geq 0, w_i \geq 0 \tag{34}$$
$$\phi^T W[u - \phi] \geq 0 \tag{35}$$

where

$$\phi(\cdot) = \begin{bmatrix} \phi_1(\cdot) \\ \vdots \\ \phi_N(\cdot) \end{bmatrix} \quad u = \begin{bmatrix} u_1 \\ \vdots \\ u_N \end{bmatrix} \tag{36}$$

and $W = \text{diag}(w_1, \ldots, w_N)$. The following standing assumptions are made throughout the paper

Assumption 1: The size of the MAS must be $N \geq 2$.

Assumption 2: The multi-agent system must admit (strongly) connected graph, with the minimum and maximum of the Laplacian matrix, $\mathcal{L} \in \mathbb{R}^{N\times N}$ being zero and non-zero respectively.

Assumption 1 is entirely logical ($N < 2$ would not constitute a network). Assumption 2 is a common assumption in the study of networked control system; it is possible it could be relaxed but this is not explored in this paper

The control law for the MAS is given by the relative state-feedback:

$$u_i = K_i z_i \tag{37}$$

With relative state information of each agent:

$$z_i = \sum_{j\in\mathcal{N}} (\xi_i - \xi_j) \tag{38}$$

where the dynamics of each agent is described by ξ

For the agent i, based on the above relative output information, the following consensus algorithm control protocol is designed:

$$u_i = K \sum_{j\in\mathcal{N}_i} a_{ij}(\xi_i - \xi_j) \tag{39}$$

as the graph is unweighted and $a_{ij} = 1$

$$u_i = K \sum_{j\in\mathcal{N}_i} (\xi_i - \xi_j) \tag{40}$$

where K is the state feedback matrix to be determined.

$$K = LW^{-1} \tag{41}$$

The problem can be re-defined for the case of a MAS by defining:

$$X(t) := \begin{bmatrix} x_1(t) \\ \vdots \\ x_N(t) \end{bmatrix} \; U(t) := \begin{bmatrix} u_1(t) \\ \vdots \\ u_N(t) \end{bmatrix} \; \Phi(t) := \begin{bmatrix} \phi_1(u_1(t)) \\ \vdots \\ \phi_N(u_N(t)) \end{bmatrix} \tag{42}$$

It follows that

$$\dot{X}(t) = (I_N \otimes A)X(t) + (I_N \otimes B)\Phi(U(t)) \tag{43}$$
$$U(t) = (\mathcal{L} \otimes K)X(t) \qquad \mathcal{L} \geq 0 \tag{44}$$

Using graph theory, the communication network can be represented by the Laplacian matrix and its spectral decomposition

$$\mathcal{L} = \mathcal{D} - \mathcal{A} = V\Lambda V^T \tag{45}$$

with positive semi-definite matrix $\Lambda = diag(\lambda_1, ..., \lambda_N)$ and orthogonal matrix V such that $A \times V = V \times D$.

The problem to be solved is then the following:

Problem 1: Consider system (43) and (44) where the system matrices A, B and the Laplacian $\mathcal{L}$ are given. Find a relative state-feedback gain K such that the origin of (43) is asymptotically stable.

B. Problem Solution

Since the system (43)-(44) is in precisely the same form as the Lur'e system discussed in Section II-C, it would be possible to formulate an LMI-based design procedure for K in a relatively straightforward manner. However, a brute-force application of these methods would be computationally inefficient for large N since the number of decision variables grows, roughly as a quadratic in N and the time required to solve an LMI grows, roughly as a 5th order polynomial function of N. In this section Problem 1 is solved using the structure lent to the problem by its network.

The first step is to re-write the system in different coordinates, remembering that V is orthogonal:

$$\tilde{X}(t) = (V^T \otimes I)X(t) \tag{46}$$

The system (43) and (44) can therefore be written as

$$\dot{\tilde{X}}(t) = (I_N \otimes A)\tilde{X}(t) + (V^T \otimes B)\Phi(U(t)) \tag{47}$$
$$U(t) = (\mathcal{L}V \otimes K)\tilde{X}(t) \qquad \mathcal{L} \geq 0 \tag{48}$$

From equation (45),

$$\mathcal{L}V = V\Lambda \tag{49}$$

and the scaling nonlinearity:

$$\Phi(U(t)) = V\,\underbrace{V^T\Phi(V}_{\tilde{\Phi}}\,\underbrace{V^T U(t))}_{\tilde{U}(t)} \tag{50}$$

The system state equation can be rewritten as:

$$\dot{\tilde{X}}(t) = (I_N \otimes A)\tilde{X}(t) + (V^T \otimes B)VV^T\Phi(VV^T U(t)) \tag{51}$$
$$= (I_N \otimes A)\tilde{X}(t) + (V^T \otimes B)V\tilde{\Phi}(\tilde{U}(t)) \tag{52}$$
$$= (I_N \otimes A)\tilde{X}(t) + (I_N \otimes B)\tilde{\Phi}(\tilde{U}(t)) \tag{53}$$

and

$$\tilde{U}(t) = V^T U(t) \tag{54}$$
$$= (V\Lambda \otimes K)V^T\tilde{X}(t) \tag{55}$$
$$= (\Lambda \otimes K)\tilde{X}(t) \tag{56}$$

The nonlinearity $\tilde{\Phi}(.)$ is bounded by the sector bound

$$\tilde{\Phi}(.) \in Sector[0, I] \tag{57}$$

if and only if the nonlinearity $\Phi(.)$ is bounded by the sector bound

$$\Phi(.) \in Sector[0, I] \tag{58}$$

and there exists diagonal matrix, W and U such that

$$V^T W V = U > 0 \tag{59}$$

with orthogonal matrix V (45).

Remark: This condition holds for an arbitrary V, given the network characteristics is that of an undirected graph. A (balanced) directed graph has complex Laplacian eigenvalues, resulting to a complex matrix eigenvector. Therefore, V will no longer be orthogonal, but unitary, and the condition III-B no longer holds.

Via standard LMI procedure

$$\begin{bmatrix} (I_N \otimes A)^T \bar{P} + \bar{P}(I_N \otimes A) & \bar{P}(I_N \otimes B) + (\Lambda \otimes K)^T W \\ * & -2W \end{bmatrix} < 0 \tag{60}$$

for some positive diagonal matrix W and positive definite matrix $\bar{P} = (I \otimes P)$, the above LMI can be reduced to a smaller inequality.

Using the Kronecker property

$$(A \otimes B)^T = A^T \otimes B^T$$

and the product of two Kronecker products

$$(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$$

it can be seen that the term

$$\tilde{P}(I_N \otimes A) = (I_N \otimes P)(I_N \otimes A) = (I_N \otimes PA) \tag{61}$$

and therefore the (1,1) element

$$(I_N \otimes A)^T \tilde{P} + \tilde{P}(I_N \otimes A) = I_N \otimes (A^T P + PA) \tag{62}$$

Also, since Λ is diagonal, expanding the (1,2) element of inequality (III-B) gives

$$\bar{P}(I_N \otimes B) + (\Lambda \otimes K^T)W = \begin{bmatrix} PB & & \\ & \ddots & \\ & & PB \end{bmatrix} + \begin{bmatrix} \lambda_1 K^T W & & \\ & \ddots & \\ & & \lambda_m K^T W \end{bmatrix} \tag{63}$$

Therefore using congruence transformations to permute the rows/columns of inequality (III-B), it follows that this inequality is solved if the N smaller inequalities are satisfied:

$$\begin{bmatrix} A^T P + PA & PB + \lambda_i K^T W \\ B^T P + \lambda_i W K & -2W \end{bmatrix} < 0 \quad \forall i \in \{1, ..., N\} \tag{64}$$

with λ_i corresponding to the eigenvalues of the Laplacian matrix $\mathcal{L}$, and matrix W being a scalar value. The matrix can be rewritten by defining $L^T = K^T W$ with gain feedback matrix $K = LW^{-1}$

$$\begin{bmatrix} A^T P + PA & PB + \lambda_i L^T \\ B^T P + \lambda_i L & -2W \end{bmatrix} < 0 \quad \forall i \in \{1, ..., N\} \tag{65}$$

Resulting to an LMI in P, L and W with only one LMI variable per element.

Remark: Solving the N LMI's above therefore solves Problem 1 and it can be seen that this scales linearly with network size; that is the number of LMI's which needs to be solved is proportional to the size of the network N. This is better than the brute force approach mentioned earlier. □

With the above in mind, the following is the main result of the paper.

Theorem 1: Let Assumptions 1 and 2 be satisfied. Consider the system (43)-(44), where $\Phi(\cdot)$ is a decentralised nonlinearity and such that $\Phi \in \text{Sector}[0, I]$. Assume further there exist a positive definite matrix $P > 0$, a positive definite diagonal matrix $W > 0$ and an unstructured matrix L of appropriate dimensions satisfying the following matrix inequalities

$$\begin{bmatrix} A^T P + PA & PB + \lambda_i L^T \\ B^T P + \lambda_i L & -2W \end{bmatrix} < 0$$

for $\lambda_i = \lambda_{min}$ and $\lambda_i = \lambda_{max}$ where these scalars are, respectively, the minimum and maximum eigenvalues of of the Laplacian matrix $\mathcal{L}$, then the origin of the system (43) is globally asymptotically stable when the feedback gain K is chosen as $K = LW^{-1}$

Proof: The proof is effectively the first part of this subsection. It remains to prove that the N inequalities (65) can be replaced by simply two inequalities. To see this is indeed the case, note inequality (65) can be written as, for each i

$$\begin{bmatrix} A^T P + PA & PB \\ B^T P & -2W \end{bmatrix} + \lambda_i \begin{bmatrix} 0 & L^T \\ L & 0 \end{bmatrix} < 0 \tag{66}$$

Now, if inequality (65) is satisfied, or equivalently (66) is satisfied for $\lambda_i = \lambda_{min}$ and $\lambda_i = \lambda_{max}$, then it follows by convexity that inequality (66) will be hold for all $\lambda_i \in [\lambda_{min}, \lambda_{max}]$; therefore it is sufficient to check inequality (66) at the end points of the interval and hence it is sufficient to solve the two inequalities in Theorem 1.

The above theorem is significant because, it means only two LMI's need to be solved *regardless of the size of the network* and thus the scalability of the result is appealing.

IV. MATLAB IMPLEMENTATION

The *Appendix* describes the design procedure for the implementation of the Lur'e MAS and LMIs. It includes the `MATLAB` code, which is used to implement the four agent system. The components are then plotted to explore the possibility of consensus. The four agent system is modeled using the Lur'e dynamics. The three state components of each agent are then plotted to explore the possibility of consensus. The stability test implements LMIs testing the smallest and largest eigenvalue of the Laplacian matrix. The first LMI tests the smallest eigenvalue, and takes form of equation (65):

$$\begin{bmatrix} A^T P + PA & PB + \lambda_{min} L^T \\ B^T P + \lambda_{min} L & -2W \end{bmatrix} < 0 \qquad (67)$$

and similarly, the second LMI tests the largest eigenvalue of the Laplacian matrix:

$$\begin{bmatrix} A^T P + PA & PB + \lambda_{max} L^T \\ B^T P + \lambda_{max} L & -2W \end{bmatrix} < 0 \qquad (68)$$

With the third condition,

$$P \geq I \qquad (69)$$

A simple Simulink block diagram models the Lur'e MAS with the state-feedback and a saturation nonlinearity at the input (since the nonlinearity is bounded in Sector $[0, I]$).

A. Simulation Example

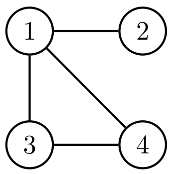

Fig. 3. Communication network graph

Consider a Multi-agent Lur'e system consisting of four agents ($N = 4$), and undirected topology as shown in Figure 3. The Laplacian matrix for the communication network is

$$\mathcal{L} = \begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 2 & -1 \\ -1 & 0 & -1 & 2 \end{bmatrix}$$

with the Laplacian eigenvalues

$$\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 3 \\ 4 \end{bmatrix}$$

For the state space realisation of the nonlinear system

$$A = \begin{bmatrix} -2 & 1 & 0 \\ 1 & -1 & 1 \\ 0 & -2 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 0 \\ 3 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}, D = 0$$

Section II is used to simulate this example and produce the following results.

B. Results

The states of each agent in the MAS are shown in Figure 4, Figure 5 and Figure 6 respectively. It can be seen from the graph that all three states of an agent can achieve consensus.

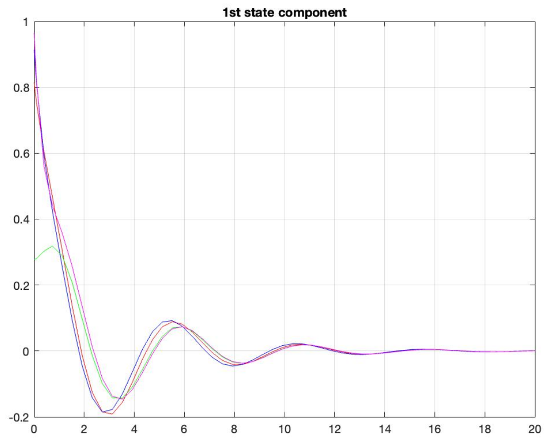

Fig. 4. First state component, x_{i1}, $i = 1, 2, 3, 4$

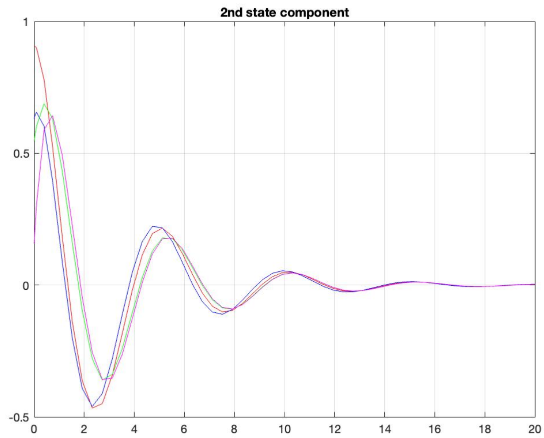

Fig. 5. Second state component, x_{i2}, $i = 1, 2, 3, 4$

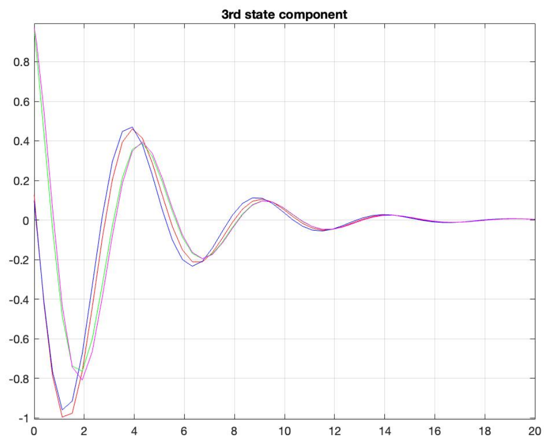

Fig. 6. Third state component, x_{i3}, $i = 1, 2, 3, 4$

The code II printed the following:

```
  lambda_min =
  0

  lambda_max =
  4

  Solver for LMI feasibility problems
L(x) < R(x)
   This solver minimizes t subject to
L(x) < R(x) + t*I
   The best value of t should be
negative for feasibility

  Result: best value of t: -0.025080
          f-radius saturation: 0.000% of
R = 1.00e+09

  Agent 1 stable
  Agent 2 stable
  Agent 3 stable
  Agent 4 stable
```

The inequality (65) holds for $\lambda_{1,2,3,4}$ and therefore verifies the stability of each agent. The stability test uses LMIs for $\lambda_{min} = 0$ and $\lambda_{max} = 4$ of the given communication network- the two extreme values of the Laplacian eigenvalue.

V. Conclusion

This paper has proposed a new scalable algorithm for the design of state-feedback controllers for strongly connected networked control systems. The main results show that, under logical conditions, only two linear matrix inequalities need to be solved to obtain a globally stabilising state feedback gain, *regardless of the size* of the network. To the authors' best knowledge this is an entirely new technical result.

The original aim of this work was to investigate consensus of multi-agent networks. As a stepping stone to this, stability results for Lur'e type MAS have been obtained. Since the consensus problem, roughly amounts to a change of state coordinates, much of the work needed to prove consensus has been done. However, additional information is required, introducing the slope of the nonlinearity as discussed in [16] The state-feedback gain K was designed to stabilise each agent (in modified co-ordinates) of the network only using the two extreme eigenvalues of the Laplacian matrix. There are numerous possible extensions to this work:

- To investigate the design of state-feedback controllers K to minimise the cost function (which could be at the agent-level or the system level)
- To investigate the use of alternative control laws or different cost functions
- To investigate different assumptions on the individual agents and also the network protocol.
- To investigate the robustness of such algorithms to uncertainty, time-delays and such like.

These, and other similar areas of research, constitute interesting topics which will expand the applicability of control theory to real networked control systems.

Appendix I
Design Procedure

This section aims to provide an overview of the project's learning process and the general design procedure. It also provides justifications for the various considerations involved in the project. The method of approach was a direct one-to-one translation of theory and calculations into code.

A. *network_build.m*

This code builds the network, calls the state-feedback design file and simulates the non-linear MAS, then relating to the consensus of the weakly nonlinear system.

- Lines 1-7: The connections of the Multi-Agent System (MAS), via graph theory, can be described using the Laplacian matrix. The Laplacian matrix is equivalent to the Adjacency Matrix subtracted from the Degree Matrix.
- Lines 9-13: Implements the state space realisation of the system given and declares G is the system created.
- Lines 15-21: The Laplacian matrix has N eigenvalues, with the minimum eigenvalue being 0 as all strongly connected graphs have this property. The maximum eigenvalue of the Laplacian matrix is defined, alongside the minimum eigenvalue. Then implementing the function created via `'stability_test_LMI'`, which tests the stability of the MAS with the connections specified.
- Lines 23-31: The matrix `'Mbig'` is associated with all the eigenvalues of the Laplacian matrix, `'Lambda_i'`. This test is to prove if the LMIs hold for both `'Lambda_min'` and `'Lambda_max'`, stability should hold for all `i`.
- Lines 33-45: Implements the new state space system defined as a closed loop network has the following dynamics:

$$\begin{cases} \dot{X}(t) &= \underbrace{(I_N \otimes A)}_{A_{new}} X(t) + \underbrace{(I_N \otimes B)}_{B_{new}} \Phi(u) \\ U &= \underbrace{(\mathcal{L} \otimes K)}_{K_{new}} X \qquad \mathcal{L} \geq 0 \end{cases}$$

  The position of each agent is set to random, and as each agent has three state components, and there are four agents total, the vector describing the position is a 12×1 vector. The time the simulation should be ran is defined, as well as simulating the Simulink file `'network_sim'` which uses variables from the workplace of this file. Indicating once this file has been ran, the simulation can then be ran.
- Lines 47-78: Plots the first state component x_1, second state component x_2 and third state component x_3 of each agent in the Multi-agent system. The plot can be

used to determine whether or not the weakly non-linear system can achieve consensus.

B. *stability_test_LMI.m*

This MATLAB file designs a state feedback gain K, which is a stabilising gain feedback. This means the gain stabilises each agent of the network, by using the defined Laplacian matrix of communication network.

- Line 1: Declares a function named `'stability_test_LMI'` that accepts inputs `G,lambda_min,lambda_max` and returns outputs `tmin,K,data`. This declaration statement is the first executable line of the function.
- Lines 3-4: Extracts the matrix data `A, B, C, D` from the state-space model `G` from the `network_build.m` file. Defines `np` and `m` as the returning lengths of the dimensions of `B`.
- Lines 6-12: Before starting the description of a new LMI system, the internal representation must be initialized. The structure and size of the LMI variables are defined using the `X = lmivar(type,struct)` syntax. The new matrix variables `P, L` and `W` in the LMI system are defined.
- Lines 14-40: The first LMI tests the smallest eigenvalue of the Laplacian matrix, implementing equation (67). Similarly, the second LMI tests the largest eigenvalue of the Laplacian matrix, implementing equation (68) With the third condition implementing equation (69). Then obtaining the internal representation of the LMI system.
- Lines 41-42: The solution `xfeas` of the defined LMI system `LMISYS1` is a particular value of the decision variables for which all LMIs are satisfied.
- Lines 44-55: Uses `dec2mat` to extract the matrix variable value from the vector of decision variable `xfeas`, extracting `Popt, Lopt` and `Wopt`. Recall, if there exists a positive semi definite matrix P satisfying the LMI, a feedback gain matrix is given by $K = LW^{-1}$.

C. *network_sim.slx*

This Simulink file models the Lur'e system with the state-feedback and a saturation nonlinearity at the input.
The MATLAB files work together, and the Simulink file uses the control toolbox and the variables created on the workplace. The model can only be run after the `'network_build.m'` file as it requires variables from the MATLAB workplace. The outputs from this model are then extracted and plotted on the same file.
To model the nonlinearity for a specified sector, the 'Dead zone' block is used as it generates zero output within a specified region, called its dead zone. The lower limit (0) and upper limit (1) of the dead zone are specified as the Start of dead zone and End of dead zone parameters.

Appendix II
MATLAB/ Simulink Implementation

A. *network_build.m*

```
%% Network
% Adjacency matrix
Adj =[ 0 1 1 1; 1 0 0 0 ; 1 0 0 1; 1 0 1 0];
% Degree matrix
Deg = [ 3 0 0 0; 0 1 0 0 ; 0 0 2 0; 0 0 0 2];
% Laplacian matrix= Degree-Adjacency
Lap = Deg-Adj;

%% State space realisation of agents
A = [-2 1 0; 1 -1 1; 0 -2 0];
B = [0 ; 0 ; 3];
C = [1 0 0];
G = ss(A,B,C,0);  % state-space object of agent

%% Generate stabilising state-feedback

eig_Lap    = eig(Lap);
lambda_min = 0 % (min eigenvalue of Laplacian)
lambda_max = max(eig_Lap) % eigenvalues always
    >= 0

[tmin,K,data] = stability_test_LMI(G,lambda_min
    ,lambda_max);

%% Test stability for all agents

for id=1:size(Lap,1)
    Mbig{id} = [A'*data.P+data.P*A data.P*B+
        eig_Lap(id)*K';
            (data.P*B+eig_Lap(id)*K')' -2*data.
                W];
    if max(eig(Mbig{id})) < 0
        disp(['Agent ' num2str(id) ' stable']);
    end
end

%% Simulations

Anew = (kron(eye(4),A));
Bnew = (kron(eye(4),B));
Knew = (kron(Lap,K));

% Agent initial position (random)
x0=rand(12,1);

% Stop time of simulation
tstop = 20;

sim('network_sim');

%% Plot
x = ans.x;
% first state component of all agents
figure;
plot(x.time,x.signals.values(:,1),'r');
hold on
plot(x.time,x.signals.values(:,4),'b');
plot(x.time,x.signals.values(:,7),'g');
plot(x.time,x.signals.values(:,10),'m');
grid on
title('1st state component')
hold off
% second state component of all agents
figure;
plot(x.time,x.signals.values(:,2),'r');
hold on
plot(x.time,x.signals.values(:,5),'b');
plot(x.time,x.signals.values(:,8),'g');
plot(x.time,x.signals.values(:,11),'m');
```

```
grid on
title('2nd state component')
hold off
% third state component of all agents
figure;
plot(x.time,x.signals.values(:,3),'r');
hold on
plot(x.time,x.signals.values(:,6),'b');
plot(x.time,x.signals.values(:,9),'g');
plot(x.time,x.signals.values(:,12),'m');
grid on
title('3rd state component')
hold off
```

B. *stability_test_LMI.m*

```
function [tmin,K,data] = stability_test_LMI(G,
    lambda_min,lambda_max)

[A,B,C,D] = ssdata(G);
[np,m]    = size(B);

%% Set up LMIS
setlmis([])

% Specify the structure and size of LMI
    variables
P   = lmivar(1,[np,1]);
L   = lmivar(2,[m,np]);
W   = lmivar(1,[1,0]*m);

% First LMI
%first term A'P+PA
lmiterm ([1 1 1 P],1,A, 's');
%second term PB
lmiterm ([1 1 2 P],1, B) ;
lmiterm ([1 1 2 -L],lambda_min ,1);
%third term B'P
lmiterm ([1 2 1 P],B',1) ;
%fourth term -2W
lmiterm ([1 2 2 W],-2 ,1);

% Second LMI
%first term A'P+PA
lmiterm ([2 1 1 P],1,A, 's');
%second term PB+L'
lmiterm ([2 1 2 P],1, B) ;
lmiterm ([2 1 2 -L],lambda_max ,1);
%third term B'P+L
lmiterm ([2 2 1 P],B',1) ;
lmiterm ([2 2 1 L],1, 1) ;
%fourth term -2W
lmiterm ([2 2 2 W],-2 ,1);

% Third LMI
lmiterm([3,1,1,P],-1,1);

LMISYS1 = getlmis;
% check the feasibility
[tmin,xfeas] = feasp(LMISYS1);

%% Recover decision variables and generate
    output data

Popt=dec2mat(LMISYS1, xfeas, P);
Lopt=dec2mat(LMISYS1, xfeas, L);
Wopt=dec2mat(LMISYS1, xfeas, W);
K=Lopt*inv(Wopt);

data.P = Popt;
data.L = Lopt;
data.W = Wopt;

end
```

C. *network_sim.slx*

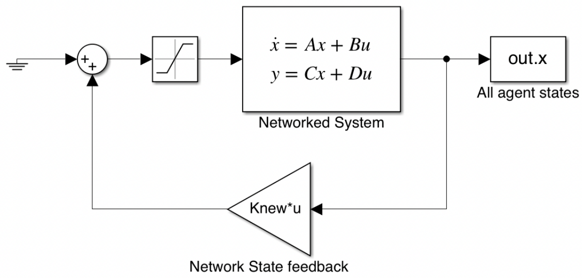

Fig. 7. Simulink model of Lur'e System

REFERENCES

[1] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, "Linear Matrix Inequalities in System and Control Theory," *SIAM studies in applied mathematics*: 15, 1994.

[2] H. Khalil, "Nonlinear Systems," 3rd Edition, *Prentice-Hall, Upper Saddle River*, 2002.

[3] G. Duan and H. Yu, "LMIs in Control Systems - Analysis, Design and Applications". 1st ed. 2013

[4] S. S. Kia, B. Van Scoy, J. Cortes, R. A. Freeman, K. M. Lynch and S. Martinez, "Tutorial on Dynamic Average Consensus: The Problem, Its Applications, and the Algorithms", *IEEE Control Systems Magazine* pp. 40-72 (2018)

[5] S. Liu, J. Sun, H. Zhang and M. Zhai, "Coordination for Lur'e Multi-Agent Systems: Fully Distributed Event-Driven Approach With Single-Event Monitoring Condition," *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2022

[6] S. Pettersson and B. Lennartson, "An LMI approach for stability analysis of nonlinear systems,"

[7] Z. Li, Z. Duan, and G. Chen, "Global synchronised regions of linearly coupled Lur'e systems," *Int. J. Control*, vol. 84, no. 2, pp. 216-227, 2011. *European Control Conference (ECC)* pp. 2317-2322, 1997

[8] Y. He and M. Wu, "Absolute stability for multiple delay general Lur'e control systems with multiple nonlinearities," *Journal of Computational and Applied Mathematics*, 159 pp. 241–248, 2003.

[9] J. W. Brewer. "Kronecker Products and Matrix Calculus in System Theory," *IEEE Transactions on Circuits and Systems*, 25-9:772–781, 1978.

[10] A. Marsden. "Eigenvalues of the Laplacian and their relationship to the connectedness of a Graph," *University of Chicago REU Program*, 2013

[11] V. M. Popov, "Absolute stability of nonlinear systems of automatic control," *Automation and Remote Control*. vol. 22, no. 8, pp. 857–875, 1961.

[12] T. Puller, A. Lecchini-Visintini and M. C. Turner, "On the design of an observer for the detection of null-current shifts in an electro-hydraulic actuation system," *2018 European Control Conference (ECC), pp. 697-702*, 2018.

[13] M. Arcak and P. Kokotovic, "Observer-based control of systems with slope-restricted nonlinearities," *IEEE Transactions on Automatic Control, vol. 46, no. 7, pp. 1146-1150*, 2001.

[14] P. P. Menon, A. Seuret and C. Edwards, "Consensus control laws with guaranteed performance in presence of fixed delays," *2014 European Control Conference (ECC)*, pp. 874-879, 2014.

[15] G-P. Liu, "Consensus and Stability Analysis of Networked Multiagent Predictive Control Systems," *IEEE Transactions on Cybernetics*, vol. 47, no. 4, pp. 1114-1119, 2017.

[16] X. Liu, J. Du and Q. Gao. "Stability Analysis of Nonlinear Systems with Slope Restricted Nonlinearities," *The Scientific World Journal*. 278305. 2014.

[17] C. Böhm, S. Yu, R. Findeisen and F. Allgöwer, "Predictive control for lure systems subject to constraints using LMIs," *2009 European Control Conference (ECC)*, pp. 3389-3394, 2009.