UPPSALA
UNIVERSITET

UPTEC E 20012
Examensarbete 30 hp
Juni 2020

Distribution of Control Effort in Multi-Agent Systems

Autonomous systems of the world, unite!

Magnus Axelson-Fisk

Teknisk- naturvetenskaplig fakultet
UTH-enheten

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
http://www.teknat.uu.se/student

Abstract

Distribution of Control Effort in Multi-Agent Systems

Magnus Axelson-Fisk

As more industrial processes, transportation and appliances have been automated or equipped with some level of artificial intelligence, the number and scale of interconnected systems has grown in the recent past. This is a development which can be expected to continue and therefore the research in performance of interconnected systems and networks is growing. Due to increased automation and sheer scale of networks, dynamically scaling networks is an increasing field and research into scalable performance measures is advancing.

Recently, the notion gamma-robustness, a scalable network performance measure, was introduced as a measurement of interconnected systems robustness with respect to external disturbances. This thesis aims to investigate how the distribution of control effort and cost, within interconnected system, affects network performance, measured with gamma-robustness. Further, we introduce a notion of fairness and a measurement of unfairness in order to quantify the distribution of network properties and performance. With these in place, we also present distributed algorithms with which the distribution of control effort can be controlled in order to achieve a desired network performance.

We close with some examples to show the strengths and weaknesses of the presented algorithms.

Handledare: Steffi Knorn
Ämnesgranskare: Mikael Sternad
Examinator: Mikael Bergkvist
ISSN: 1654-7616, UPTEC E 20012
Tryckt av: Uppsala

Populärvetenskaplig sammanfattning

I och med att fler och fler system och enheter blir utrustade med olika grader av intelligens så växer både förekomsten och omfattningen av sammankopplade system, även kallat Multi-Agent Systems. Sådana system kan vi se exempel på i traffikledningssystem, styrning av elektriska nätverk och fordonståg, vi kan också hitta fler och fler exempel på så kallade sensornätverk i och med att Internet of Things och Industry 4.0 används och utvecklas mer och mer. Det som särskiljer sammankopplade system från mer traditionella system med flera olika styrsignaler och utsignaler är att dem sammankopplade systemen inte styrs från en central styrenhet. Istället styrs dem sammankopplade systemen på ett distribuerat sätt i och med att varje agent styr sig själv och kan även ha individuella mål som den försöker uppfylla. Det här gör att analysen av sammankopplade system försvåras, men tidigare forskning har hittat olika regler och förhållninssätt för agenterna och deras sammankoppling för att uppfylla olika krav, såsom stabilitet och robusthet.

Men även om dem sammankopplade systemen är både robusta och stabila så kan dem ha egenskaper som vi vill kunna kontrollera ytterligare. Specifikt kan ett sådant prestandamått vara systemens motståndskraft mot påverkan av yttre störningar och i vanliga olänkade system finns det en inneboende avvägning mellan kostnad på styrsignaler och resiliens mot yttre störningar. Samma avvägning hittar vi i sammankopplade system, men i dessa system hittar vi också ytterligare en dimension på detta problem. I och med att ett visst mått av en nätverksprestanda inte nödvändigtvis betyder att varje agent i nätverket delar samma mått kan agenterna i ett nätverk ha olika utväxling mellan styrsignalskostnad och resiliens mot yttre störningar. Detta gör att vissa agenter kan ha onödigt höga styrsignalskonstander, i den mening att systemen skulle uppnå samma nätverksprestanda men med lägre styrsignalskostnad om flera av agenterna skulle vikta om sina kontrollinsatser.

I det här examensarbetet har vi studerat hur olika val av kontrollinsats påverkar ett sammankopplat systems prestanda. Vi har gjort detta för att undersöka hur autonoma, men sammankopplade, agenter kan ändra sin kontrollinsats, men med bibehållen nätverksprestanda, och på det sättet minska sina kontrollkostnader. Detta har bland annat resulterat i en distruberad algoritm för att manipulera agenternas kontrollinsats så att skillnaderna mellan agenternas resiliens mot yttre störningar minskar och nätverksprestandan ökar. Vi avslutar rapporten med att visa ett par exempel på hur system anpassade med hjälp av den framtagna algoritmen får ökad prestanda. Avslutningsvis följer en diskussion kring hur vissa antaganden kring systemstruktur kan släppas upp, samt kring vilka områden framtida forskning skulle kunna fortsätta med.

First and foremost I would like to give my sincerest thanks to advisor Prof. Steffi Knorn, Otto-von-Guericke University, who has been of tremendous help and encouragement throughout this work, I would also like to extend my gratitude to Prof. Mikael Sternad, for valuable input as subject reader. Further, I would also like to thank Annett Wertran, Otto-von-Guericke University, for great help with any and all formal matters.
Additionally, I would like to thank my friends at Harnackstrasse 6, who has been of significant in maintaining some normality, sanity and work moral throughout this spring.

Last, but not least, I would like to thank all my friends and family in Uppsala for checking up on and encouraging me.

Contents

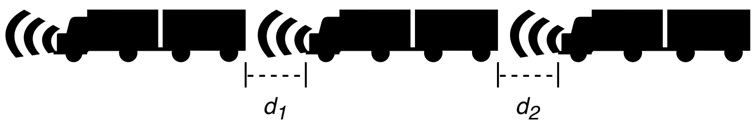


Figure 1: Example of three heavy-duty vehicles platooning in order to keep inter vehicular distance, d_1 and d_2, to some predefined values.

1 Introduction

As the research on industrial and home automation, self-driving vehicles and artificial intelligence grows, so does the number and scale of autonomous agents. With this, the development of interconnected systems emerge, where agents cooperate in a given environment to pursue either individual or common goals, such as balancing power load in an electrical grid, moving through a physical environment or reaching a consensus on a measurement. An agent can, in these systems, be described as an entity, which is able to control itself, either partially or fully, in order to pursue some defined goal. As this is a very broad definition of agent, the applications of agents and Multi-Agent Systems, MAS, are equally broad. The MAS are built up by the interconnections of these agents and it should be noted that these connections do not necessarily only model physical interaction between the agents, but could also model information exchange, leading to cyber-physical interactions.

This means that there are few restrictions in what can be modelled or controlled as a MAS, and the examples range from control of electrical grids [1], traffic management systems [2] and vehicle control [3] to communication networks [4] and sensor networks [5], to name a few. As MAS, or large-scale interconnected systems, are built up by many autonomous agents, there is often no central controlling entity. Generally this may also mean that there is no entity that has knowledge of the entire system, rather that each agent has access to only the knowledge of itself and that of its neighbouring agents, which may be beneficial since the computational load may be spread among the agents.

One use of interconnected systems, that has received a lot of attention, is within vehicle platooning, a representation of this can be seen in Fig. 1, and in [6] it was shown that given a platoon of only three vehicles, one leader and two following vehicles, the fuel savings were substantial[1]. The idea of vehicle platooning is maybe not new, but with the expanding research and legislation regarding autonomous vehicles, this can certainly be an area in which there are significant environmental costs to be saved in the near future. Furthermore, given such a platoon, there may be an uneven distribution of control effort leading to an inefficient control, such that some vehicles are forced to compensate for other vehicles in the platoon, with faster acceleration and deceleration. If such unnecessary control efforts and cost can be decreased, without deteriorating the overall performance of the vehicle platoons, the savings can be made even larger.

1.1 Background

As MAS are consisting of interconnected *autonomous* agents, the control of networks is mostly decentralised, which means that the agents control their own actions and need to cooperate in order to pursue a individual or shared goal. However, as the agents are influencing their

[1]At Scania there are currently projects on automatisation of vehicles and platooning, see for instance https://www.scania.com/group/en/home/newsroom/news/2018/automated-platooning-step-by-step.html for a video of how this may look.

neighbours, networks and decentralised control can be configured such that the agents increase the stability, or with a faulty configuration, or in the case of malicious agents, destabilise the networks.

Due to the nature of MAS, their analysis is non-trivial as the stability, and other desired properties of a system, need to be ensured in a decentralised manner, i.e. without a centralised control which has knowledge of the entire system. This has been done in for instance [7] and [8], where the stability of large-scale interconnected systems was studied. Another aspect of system performance is that of robustness, which is an important aspect of large-scale interconnected systems as disturbances may grow as they propagate through the system, not necessarily leading to instability in a control theoretic sense, but could cause cascaded failures of the subsystems or large performance loss. One such performance measure is the notion of *string stability*, introduced in [9], which is motivated by vehicle platoons and requires that the disturbances do not grow as they propagate through the network, the use of this can be found in e.g. [6]. In [10] a performance measure of a systems ability to reject external disturbances was introduced, denoted as *γ-robustness*. The authors of [10] show that the performance of a network, with respect to γ-robustness, generally deteriorates as the amount and weights of incoming edges increases. Further, the authors present conditions on how to adapt control parameters, such that structural changes in a network, i.e. adding and removing agents and edges, can be made without deteriorating the disturbance rejection capability.

These conditions lead in general to an increase of agents' control efforts, thereby control costs, in order to achieve the same bound γ, as the incoming edges increase. However, due to the dynamic interconnections of a system, there may exist multiple, if not infinitely many, combinations of control parameters that lead to the same given bound γ. This, in turn, can mean that some agents have an unnecessarily large control effort, such that the same γ could be met with a lower control effort if some other agents increased their control effort. In contrast, there could be agents that have lower control efforts, thereby lower control costs, but forces their neighbours to increase control costs in order to ensure the desired bound γ.

This thesis builds on [10] and extends some of the results on γ-robustness and network structure, within this we will focus on the relationship between control parameter and network performance. Further, we investigate the distribution of control effort in large-scale interconnected systems and how this affects γ-robustness and disturbance rejection capability. A similar idea was discussed in [11], where an algorithm for reweighting the edge connections between agents in order to increase disturbance rejection was presented. However, the results in this thesis do not only differ to [11] by the assumptions made on the system, but also on the intention of the algorithms. As the algorithm in [11] focus on improving disturbance rejection capability, the results in this thesis focuses on the distribution of control effort and thereby reducing unnecessary and uneven control effort.

1.2 Objectives

This project aims at developing an understanding and researching γ-robustness of systems, develop control strategies to ensure that structural changes in systems are γ-scalable. Further, we aim to research how the distribution of control efforts affects network performance, with and without constraints on systems. We intend to do this for both linear and non-linear system dynamics, and extend the results from scalar to non-scalar systems.

1.3 Contributions

In this thesis we have extended the existing results on γ-robustness, regarding how the state deviation bound behaves as a function of control parameters. Specifically we have found that for linear scalar systems, the state deviation bounds, u, is known to be a non-increasing continuous function of control parameters, the set of possible state deviation bounds, $\mathcal{U}$, of a system is now known to be convex and the individual bounds, u_i, is a convex function of control parameters, these results can be found in Section 4.3. Further, we have introduced the new notion of χ-fairness in system parameters and the measurement of distance from current parameter setting to a fair setting, which can be found in Section 5. We have also presented a decentralised algorithm which can be used to reduce unfairness in state deviation bound, u, and the overall bound γ, for linear scalar systems. The presented algorithm can be found in Section 5.5.

These results have also been submitted as part of a paper to the 59th annual Conference on Decision and Control, at this date we have received constructive criticism from the reviewers and are currently awaiting a final decision on acceptance.

Notation

We consider vector inequalities elementwise, if nothing else is noted. $\mathbb{R}$ and $\mathbb{R}_+$ denote the real and positive real numbers. Given a set $\mathcal{N}$, $|\mathcal{N}|$ denotes the number of elements in $\mathcal{N}$. $\mathbb{1}$ denotes the column-vector of all ones and $\mathbb{I}$ denotes the identity with appropriate dimension. $(A)_{ij}$ denotes the element in the ith row jth column, and $(A)_{i\bullet}$ denotes the ith row of matrix A. e_i denotes the column vector of zeros, with ith element equal to one.

2 Preliminaries

In this section we will provide some preliminaries on some of the concepts used to develop many of the results in this thesis. However, as some previously existing results are used directly in proofs, and the general structure of this master thesis, some theory may be presented during the sections that follow instead.

2.1 Linear Algebra

As many of the proofs are mathematically heavy, and may use operations and properties that might not be widely used, we will briefly introduce and state some of these operations, their properties, and properties of a few classes of matrices that will reoccur in this master thesis. The notations and properties of the operations are standard and can be found in [12].

2.1.1 Kronecker and Hadamard products

The normal matrix multiplication product of the two matrices A and B is defined as the rows of A multiplied with the columns of B, and if $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{p\times q}$, then the multiplication of the two, AB, is only defined when $n = p$. However, there exists two other matrix multiplication operations; the *Kronecker* product, denoted $\otimes$, and the *Hadamard* product, denoted $\odot$.

Kronecker product

The Kronecker product is defined for two matrices, $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{p\times q}$, as

$$A \otimes B = \begin{bmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{bmatrix},$$

which means that the dimensions of $A \otimes B$ are $mp \times nq$, due to this it is defined for any matrix dimensions $m \times n$ and $p \times q$. What follows are some of the properties of the Kronecker product.

[K_1] It is not commutative, $A \otimes B \neq B \otimes A$.

[K_2] It is associative, $A \otimes (B \otimes C) = (A \otimes B) \otimes C$.

[K_3] It is distributive, $A \otimes (B + C) = A \otimes B + A \otimes C$.

[K_4] The transpose of the Kronecker product is equal to the Kronecker product of the transposes, $(A \otimes B)^T = A^T \otimes B^T$.

For further properties and proofs, see [13].

Hadamard product

The Hadamard product is defined as the product of the element-wise multiplication of A and B, such that

$$(A \odot B)_{ij} = a_{ij}b_{ij},$$

it can also be noted that the Hadamard product is a submatrix of the Kronecker product. This means that the Hadamard product is only defined for matrices of same size, i.e. $m = p$ and $n = q$, and that the Hadamard product has the same dimensions as the original matrices. Further, the following are some of the properties of the Hadamard product

[H_1] It is commutative, $A \odot B = B \odot A$.

[H_2] It is associative, $A \odot (B \odot C) = (A \odot B) \odot C$.

[H_3] It is distributive, $A \odot (B + C) = A \odot B + A \odot C$.

[H_4] The matrix of all ones, $\mathbb{J}$, is the identity matrix under the Hadamard product, $A \odot \mathbb{J} = A$.

[H_5] The transpose of the Hadamard product is equal to the Hadamard product of the transposes, $(A \odot B)^T = A^T \odot B^T$.

For further properties and proofs, see [13].

2.1.2 M-matrices

An often occurring class of matrices, which can be found in fields such as economical, engineering and social sciences, are matrices of the form

$$A = \begin{bmatrix} a_{11} & -a_{12} & -a_{13} & \cdots \\ -a_{21} & a_{22} & -a_{23} & \cdots \\ -a_{31} & -a_{32} & a_{33} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix},$$

where all $a_{ij} \geq 0$, i.e. matrices with positive diagonal and non-positive off-diagonal elements. This class of matrices is called Z-matrix and can be written as,

$$\mathbb{Z} = \{(A)_{ij} \leq 0 \, \forall i \neq j, \, (A)_{ii} \geq 0\}.$$

Now, depending on the diagonal elements of A, matrices in the class of $\mathbb{Z}$-matrices may also be M-matrices, which is a subclass of $\mathbb{Z}$-matrices. The class of M-matrices is defined as

$$\mathrm{M} = \{A = s\mathbb{I} - B | s \geq \rho(B), B \geq 0\},$$

with $\rho(B)$ being the spectral radius of B and if s is strictly larger, $s > \rho(B)$, then A is a non-singular M-matrix. These classes of matrices have been extensively studied and we will repeat some of the most important properties here, as many of these will be used in the analysis of interconnected systems. Now, given a matrix $A \in \mathbb{Z}$, the following properties are equivalent to saying that A is a non-singular M-matrix.

[M_1] A is positive stable, i.e. the real part of all eigenvalues of A is positive.

[M_2] A is inverse positive, i.e. the inverse A^{-1} is element-wise non-negative.

[M_3] Every regular splitting of A is convergent, i.e. if $A = M - N$ then $\rho(M^{-1}N) < 1$.

[M_4] All principal minors of A are positive.

[M_5] The inverse of A, A^{-1}, has all principal minors positive.

For further properties and proofs, see for instance [14, 15].

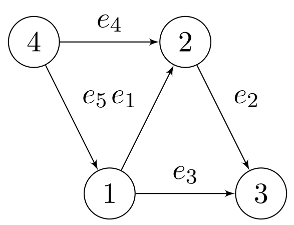


Figure 2: Graph of a directed network formed by four vertices and five edges.

2.2 Graph Theory

A very useful theoretic tool to describe the interconnections in a network is through the use of graphs. Graphs are a mathematical concept that consists of nodes, or vertices, and edges, with no inherent restrictions in what a vertex or edge can represent. Throughout this thesis we will use standard concepts of graph theory and connectivity for which the definitions can be found in [16].

In Fig. 2, a graph consisting of four vertices can be seen. As the edges between the vertices have directions this is known as a *directed* graph, if we would remove the directions it would instead be known as an *undirected* graph.

Graphs are formed by two sets characterizing the structure of the graph, one set of vertices, $V = \{v_1, v_2, \ldots, v_n\}$, and one set of edges, E, with the interconnection structure between the vertices. The graph formed by these two sets is denoted $\mathcal{G}(V, E)$. Graphs can be both undirected, where each edge (i, j) represents the connection through which both vertices can communicate, or directed, in which an edge (i, j) describes the communication from vertex j to vertex i. Regardless whether an edge has a direction or not, two vertices connected through one edge are called *adjacent* or *neighbours* and the general set of vertices connected to vertex i, each connected through only one edge, is denoted as $\mathcal{N}_i$. Now, if the graph is directed then the set $\mathcal{N}_i$ can be described to be a union of two sets, namely the set of vertices where there exists an edge (i, j), denoted $\mathcal{N}_i^{\text{in}}$, and the set of vertices where there exists an edge (j, i), denoted $\mathcal{N}_i^{\text{out}}$. Both of these sets may overlap, in the sense that a vertex j may be part of both sets, and in these cases $\mathcal{N}_i$ means the set of unique vertices. Additionally, traversing between two vertices through a set of unique edges, such that no edge is visited twice, is known as a *path* and if the two vertices are identical then this is called a *cycle* [16]. Furthermore, edges may also be weighted, such that the information communicated between two vertices is weighted accordingly to the edge, but this does not influence the directions of the edges.

2.2.1 Connectivity

As graphs can have more or less edges connecting the vertices, the graphs can also be categorized according to their *connectivity*. An undirected graph is called *connected* if there exists a path between every pair of vertices, while if there exists a directed path between every pair of vertices in a directed graph this is called *strongly connected*[2]. If there is one, or more, vertices that cannot be reached from any other vertex the graph is called *disconnected*. However, this does not mean that a directed graph which is not strongly connected is necessarily disconnected. A directed graph can be *weakly connected* if in the *disoriented graph*, the resulting graph when ignoring the directions of the edges, there exists a path between every pair of vertices. Further, a graph in which there exists an edge between every pair of vertices is called a *complete graph*.

[2]It is not necessary that the path from vertex v_i to v_j traverses through vertices in the same order, or even the same vertices, as the path from v_j to v_i for a directed graph to be strongly connected.

Additionally, an undirected graph which does not contain any cycles is said to be a *tree*, meaning that any two pair of vertices are linked by a unique path [16].

We can also define a directed graph which has the property that all other vertices can only be visited from a unique path from a single vertex r, called root, as a *spanning rooted out-branching*[17].

Definition 1. *A directed graph $\mathcal{D}$ is said to be a* spanning rooted out-branching *if it does not contain any cycles, and there exists only one vertex, r, from which there exists a unique path to all other vertices in $\mathcal{D}$.*

In Fig. 2 we can see an example of a weakly connected directed graph. We can also see that if we were to remove edges e_1 and either e_2 or e_3, there would be no cycles and from vertex 4 there would exist a unique path to all other vertices, we can therefore say that the graph in Fig. 2 contains a rooted out-branching.

2.2.2 Representing graphs through matrices

Graphs do not only have the representation through the sets of vertices and edges, they can also be represented through matrices and there exist various matrices that capture different properties of a graph.

Incidence Matrix

We can start with the *incidence* matrix, B, which captures the connections between edges and vertices and is a $m \times n$ matrix, where m is the number of edges and n is the number of vertices. For a directed graph it is defined as $B(\mathcal{D})_{ij} = 1$ if the head of edge i and vertex j are incident, $B(\mathcal{D})_{ij} = -1$ if the tail of edge j and vertex i are incident, and zero otherwise[17]. For the directed graph depicted in Fig. 2 the incidence matrix B will be

$$B(\mathcal{D})^T = \begin{bmatrix} -1 & 0 & -1 & 0 & 1 \\ 1 & -1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 \end{bmatrix}.$$

We can also define the incidence matrix for the oriented graph $\mathcal{G}$, that is where every edge has received an arbitrary direction, and denote this incidence matrix as $B(\mathcal{G}^0)$, this matrix is then defined similarly as $B(\mathcal{D})$.

Adjacency and Degree matrices

We continue with the *adjacency matrix*, A, and the *degree matrix*, Δ. These describe which vertices are neighbours, and the connections between these, and the vertices themselves, respectively. But first we need to define the representation of edges. Throughout this section, w_{ij} will be used to represent an edge from vertex j to vertex i and this weight is set as $w_{ij} = 1$ for an unweighted graph if $(i, j) \in \mathcal{E}$, or the associated weight of the edge if the graph is weighted [17]. The adjacency matrix can now be defined as

$$[A]_{ij} = \begin{cases} w_{ij} \text{ if } (i,j) \in \mathcal{E} \\ 0 \text{ otherwise.} \end{cases},$$

which means that the adjacency matrix is defined in the same manner for both undirected and directed graphs. However, the edge set, $\mathcal{E}$, may differ, which means that A_{ij} is not necessarily equal to A_{ji} for a directed graph. For an undirected graph this would be the case and the adjacency matrix would be symmetric.

We continue with the degree matrix, which is a diagonal matrix used to represent the vertices. The vertices can represented by their *in-* or *out-degree*, $d(v_i)$, which is defined as follows

$$d(v_i) = \sum_{j \in \bar{\mathcal{N}_i}} w_{ij},$$

where $\bar{\mathcal{N}_i} = \mathcal{N}_i^{\text{out}}$ if *in-degree* is used or $\bar{\mathcal{N}_i} = \mathcal{N}_i^{\text{out}}$ if *out-degree* is used, for an undirected graph $\mathcal{N}_i^{\text{in}} = \mathcal{N}_i^{\text{out}}$. This means that the degree matrix is defined as follows

$$[\Delta]_{ij} = \begin{cases} d(v_i) \text{ if } i = j \\ \quad 0 \text{ otherwise.} \end{cases}$$

Laplacian matrix

Given the definitions of the adjacency and degree matrices, we can continue with the Laplacian matrix, L. There exists various definitions of the Laplacian matrix. Here, we will only use two of these and both of these result in the same matrix for an undirected graph, but not necessarily for a directed graph. In order to separate these, we will be using $L(\mathcal{G})$ for the Laplacian formed for an undirected graph and $L(\mathcal{D})$ for the Laplacian formed for a directed graph. The Laplacian for both an undirected and a directed graph can be formed in the following way

$$L(\mathcal{D}) = \Delta(\mathcal{D}) - A(\mathcal{D}). \tag{2.1}$$

We can also see that the row sum of $L(\mathcal{D})$ will always be zero, by the definition of the degree of a node, this means that one *right* eigenvector of $L(\mathcal{D})$ will be the column vector of all ones, $\mathbb{1}$, and the corresponding eigenvalue will be equal to zero. The second definition we will use represents the Laplacian for an undirected graph $\mathcal{G}$, that represents the directed graph $\mathcal{D}$ which has been disoriented. That is, by removing the directions of the edges in the directed graph $\mathcal{D}$, the resulting undirected graph $\mathcal{G}$ is formed. The definition of the Laplacian uses the incidence matrix of the directed graph $\mathcal{D}$, or the oriented graph $\mathcal{G}^0$, as

$$L(\mathcal{G}) = B^T B. \tag{2.2}$$

It can be noted that the Laplacian for the undirected graph $\mathcal{G}$ in (2.1) is equal to the Laplacian, formed with $\mathcal{G}$, in (2.2),

$$L(\mathcal{G}) = B^T B = \Delta(\mathcal{G}) - A(\mathcal{G}).$$

With the definitions of the Laplacian in place, we continue with some of the properties of the Laplacian which we will use in the rest of the thesis. First, as the Laplacian contains information about the interconnection between nodes, the Laplacian will also contain information regarding the connectivity of the graph. We can start with noting that an eigendecomposition of $L(\mathcal{G})$ defined either as in (2.1) or (2.2), reveals that

$$\begin{aligned} \lambda_i &= v_i^T L v_i \\ &= v_i^T B^T B v_i \\ &= (Bv_i)^T (Bv_i) \end{aligned}$$

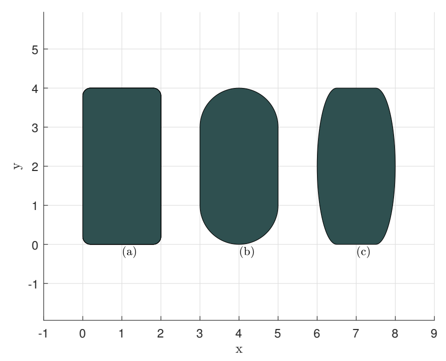
which gives that the Laplacian is positive semi-definite. As we have previously stated, one of the eigenvalues of $L(\mathcal{D})$ is equal to zero, this will in fact also be true for $L(\mathcal{G})$, as the degree of a node in $\mathcal{G}$ is defined as for a node in $\mathcal{D}$, and since $L(\mathcal{G})$ is positive semi-definite we will have

$$0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$$

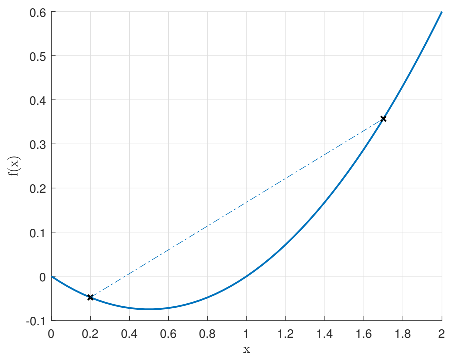
and in [17] it is shown that a graph $\mathcal{G}$ is connected if and only if

$$0 < \lambda_2.$$

2.3 Convexity



(a) Example of three convex sets, a, b and c.



(b) Example of a convex function $f(x)$ on the convex set $x \in [0, 2]$.

Figure 3: Example of convex sets and function.

A property of sets and functions that can often simplify algorithms and ensure convergence in problems is the property of convexity. A convex set is a set such that if we would draw a line between two points within the set, then all points on that line will also lie within the set. Similarly, a convex function is a function where the line between any two points on the graph of the function will lie above the function. These two definitions may be easy to verify graphically in two or three dimensions, but when this is not possible the following conditions on convex sets and functions can instead be used, [18].

A set X is convex if an only if all points on a line between two points within X are also in the set, this is equivalent as the following. Let x and y be two points such that $x, y \in X$ and let $\lambda \in (0, 1)$, then the set X is convex if and only if

$$\lambda x + (1 - \lambda)y \in X, \, \forall x, y \in X. \tag{2.3}$$

A function $f(x)$ is convex on the convex set X, then $f(x)$ is convex if and only if

$$f\left(\lambda x + (1 - \lambda)y\right) \leq \lambda f(x) + (1 - \lambda)f(y), \, \forall x, y \in X. \tag{2.4}$$

In Fig. 3 we can see both convex sets and a convex function.

2.4 Optimisation methods

In many situations there exists an optimal solution to a problem, which would solve every aspect of the said problem in the most desirable way. However, equally often this solution is not feasible, in the sense that it allocates too many resources or is reliant on near impossible performance, such that it is necessary to find the optimal solution to a problem, given some constraints. These problems are known as optimisation problems, and hence, the methods used to solve the problems are known as optimisation methods.

As there exists many different classes of constraints and problems we will only bring up a few that are relevant to the results presented in this thesis. We therefore limit the discussion to problems of the form

$$\min_{x\in X} f(x) = \frac{1}{2}x^T Mx \tag{2.5}$$

where x is vector $x \in \mathbb{R}^N$, M is a square matrix , $M \in \mathbb{R}^{N\times N}$, additionally we also assume that the set X is a convex subset of $\mathbb{R}^N$. We start by examining under which conditions a point x^* minimizes this objective function, we therefore assume that x^* minimizes $f(x)$ and that $x^* \in X$. If x^* minimizes $f(x)$ then we have that

$$f(x^*) \leq f(x), \forall x \neq x^*.$$

This shows that x^* is a critical or stationary point to $f(x)$, which gives the first order conditions. If x^* is a minimizer to $f(x)$ then

$$\nabla_x f(x^*) = 0. \tag{2.6}$$

However, this condition could also be fulfilled for some point where $f(x)$ is maximized, but if (2.5) is convex in the set X, then the first order condition will only be fulfilled for a global minimizer [19]. Now, a twice differentiable multi variable function is convex if and only if the Hessian is positive (semi-) definite [18]. We therefore examine the Hessian of $f(x)$ and find that it is

$$\nabla^2 f(x) = M. \tag{2.7}$$

We therefore assume that M is positive (semi-) definite in the following discussions, such that the objective function $f(x)$ is convex. As we have assumed that $x^* \in X$ solving this problem is known as solving an unconstrained optimisation problem. Such a problem can be solved by using the *gradient descent* method, where each iteration is found by taking a step, of suitable length, in the negative gradient direction. Choosing this step size appropriately ensures convergence, and there exists multiple conditions on how to choose a step size appropriately [19]. The iteration procedure then looks like

$$x_{k+1} = x_k - a_k \nabla f(x), \tag{2.8}$$

where a_k is an appropriately chosen step size. However, if x^* does not lie within X or there exist constraints on the solution, then we must use other techniques which we will present now.

2.4.1 Constrained optimisation

First we consider the case when the global minimizer x^* does not lie within X, but still assume that $f(x)$ is convex. One method to solve this problem is by using the *projected gradient descent* method, in which each iteration is projected onto the set. The iteration procedure thus looks like

$$x_{k+1} = P_X\left[x_k - a_k \nabla f(x)\right], \tag{2.9}$$

where P_X is the projection operator defined as

$$P_X\left[z\right] = \arg\min_{x\in X} ||z - x||. \tag{2.10}$$

In [20] it is shown that a stationary point, x_c^*, to (2.9) will satisfy the following first order condition

$$\nabla f(x_c^*)'(x_c^* - x) \geq 0,\ x \in X. \tag{2.11}$$

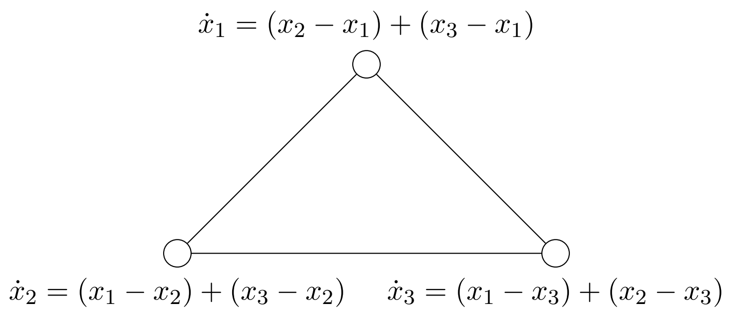

$\dot{x}_1 = (x_2 - x_1) + (x_3 - x_1)$
$\dot{x}_2 = (x_1 - x_2) + (x_3 - x_2)$
$\dot{x}_3 = (x_1 - x_3) + (x_2 - x_3)$

Figure 4: Agreement protocol on a three node network.

Further, it is shown that the iterations in (2.9) will converge to such a stationary point by choosing the step size according to a generalized Armijo rule described in [20]. However, using the projected gradient descent will only be effective if the projection onto X is not to complicated, in situations where this is not the case other methods could instead be used.

We therefore assume that we can rewrite the set, such that only vectors that fulfil some linear inequality constraints will lie within the set. The optimisation problem instead becomes

$$\begin{aligned} &\min_x f(x) = x^T M x \\ &\text{subject to } Ax \leq b. \end{aligned} \tag{2.12}$$

There exists multiple methods for solving this, and for more information about methods and conditions on optimum, such as the Karush-Kuhn-Tucker conditions, see [19].

2.5 Networks as dynamic systems

One may consider networks as a single system, and investigate the dynamics of the interconnected system. This will give us a model of which we can analyse the behaviour and performance. In the following sections, if not otherwise noted, we will consider agents as scalar systems, such that for a single agent the system dynamics can be described as

$$\dot{x}_i = -a_i x_i + \sum_{j \in \mathcal{N}_i} m_{ij} x_j, \tag{2.13}$$

where a_i is the agents' control parameter, m_{ij} is the edge weight of the edge from agent j to agent i. Given these single agent dynamics, we can describe networks and the interconnection between agents by using the standard matrices from graph theory described in Section 2.2.

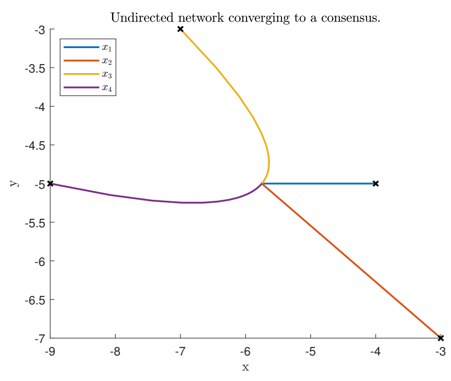
2.5.1 Agreement protocols

Many of the goals of networks, e.g. formation control, distributed estimation and sensing, can be formed to a problem of agreeing on a common value, i.e. come to a consensus. How these protocols are formed and their convergence rate is therefore of importance.

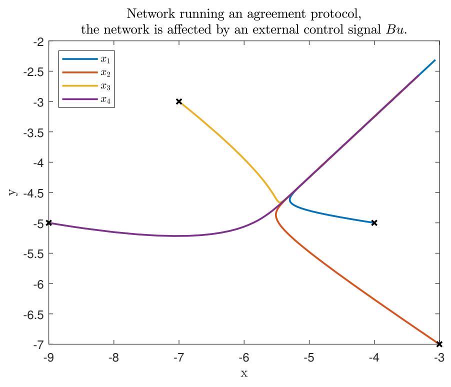
Undirected graphs

In Fig. 4 a consensus protocol over a graph, which will come to an agreement when $x_1 = x_2 = x_3$, can be seen. Mathematically this can be written as

$$\begin{aligned} \dot{x}_1 &= -2x_1 + x_2 + x_3 \\ \dot{x}_2 &= x_1 - 2x_2 + x_3 \\ \dot{x}_3 &= x_1 + x_2 - 2x_3 \end{aligned}$$



(a) Trajectories of the undirected networked formed as in (2.15), with state dynamics as in (2.14).



(b) Trajectories of the undirected networked formed as in (2.15), with state dynamics as in (2.16).

Figure 5: Examples of networks reaching consensus with and without external control signals.

which shows that the consensus protocol can be written as

$$\dot{x} = -L(\mathcal{G})x \tag{2.14}$$

by using the in-degree Laplacian of graph $\mathcal{G}$. From here it is also possible to define the agreement set, i.e. the set of states the system will converge to, and since the Laplacian has an eigenvector spanning of all ones, $v = \mathbb{1}$, with the eigenvalue zero, $\lambda = 0$, this will be the agreement set $\mathcal{A}$

$$\mathcal{A} = \text{span}\{x \in \mathbb{R}^n | x_i = x_j, \forall i, j\}\,.$$

With the agreement set in place, the convergence of the consensus protocol can be found. By noting that the solution of (2.14) with initial state $x(0) = x_0$ is

$$x(t) = e^{-L(\mathcal{G})t}x_0$$

it is possible to derive the rate of convergence. If the graphs are connected, then the Laplacian will be an M-matrix and all of its eigenvalues will be non-negative. In [17] it is shown that the eigenvalue $\lambda_1 = 0$, will have algebraic multiplicity of one for a connected graph. This gives the following relation of the eigenvalues

$$0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_n$$

and it is possible to see that it is λ_2 that gives a bound on the rate of convergence.

In Fig. 5a an example of the trajectories as a network reaches consensus can be seen. In this network, we have assumed that each agent can be represented as two decoupled states, position on x-axis and position on y-axis. This means that we treat the movement along each axis as separate networks, but with the same interconnections, such that both networks have a Laplacian formed as

$$L(\mathcal{G}) = \begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 2 & 0 \\ -1 & -1 & 0 & 2 \end{bmatrix}, \tag{2.15}$$

which is the Laplacian of the disoriented version of the graph depicted in Fig. 2. Agreement protocols can also be used in Leader-Follower applications and in Fig. 5b we can see how the agents steers through an environment. In this example a control signal, u, was added to one agent, such that the system dynamics can be described as

$$\dot{x} = -L(\mathcal{G})x + Bu, \tag{2.16}$$

where $B = [1\,0\,0\,0]^T$ and $u = 1$.

Directed graphs

For directed graphs the agreement set will be the same and analysing the convergence rate will also not change. What will change is, however, the conditions of connectivity for the agreement protocol to converge and the agreement set. For a consensus protocol to converge, (2.14) must converge, which means that the eigenvalues of $L(\mathcal{G})$ have to be non-negative and that eigenvalue $\lambda = 0$ has multiplicity one and is associated with the right eigenvector p_1. This gives the condition for a consensus protocol to converge on a directed graph is that

$$\operatorname{rank}(L(\mathcal{G})) = n - 1\,. \tag{2.17}$$

This condition on the Laplacian will be fulfilled if the graph contains a spanning rooted out-branching. A sufficient condition for this is that the graph is strongly connected. As the agreement set changes, we investigate the trajectories of the system, if we define p_1 as the right eigenvector and q_1 as the left eigenvector, normalized such that $p_1^T q_1 = 1$. We have that from any initial state x_0 the trajectories will satisfy

$$\lim_{t\to\infty} x(t) = (p_1 q_1^T)x_0. \tag{2.18}$$

Since we have $p_1 = \mathbb{1}$, this means that the trajectories will satisfy

$$\lim_{t\to\infty} x(t) = q_1^T x_0 \mathbb{1}.$$

We can therefore see that if $q_1 = c\mathbb{1}$, where c is some scalar, then the agreement protocol will converge to some weighted average. If this is not the case then the agents in the network will not converge to a consensus. This means that a directed graph will only converge to an average consensus from any initial state if $q_1 = c\mathbb{1}$ and the graph contains a spanning rooted out-branching, which is equivalent to the graph being weakly connected and balanced [17].

3 Stability in Multi-Agent Systems

Aspects of stability analysis in networks have a long history and have been studied extensively, for instance [7, 8]. The stability of a network is not necessarily reliant on the subsystems being inherently stable, instead there may exist inherently unstable subsystems but as they are interconnected they can be stabilised and the interconnected network may be stable. In turn, there may be inherently stable subsystems, which are made unstable through the interconnections. Due to this, stability in a large-scale interconnected system is a difficult problem to analyse in a decentralised fashion, we therefore start with analysing this in a centralised manner.

As an example of how connecting subsystems can render the interconnected system stable or unstable, we can consider a simple loop with a controller and a plant, as seen in Fig. 6. If we consider the situation where we have an unstable plant, we know that there exist controller designs which would stabilise the loop, in fact this is an often occurring problem solved by controllers. In turn, we also know that there may exist stable plants and controllers, which result in an unstable control loop as they are connected, due to unsuitable controller designs.

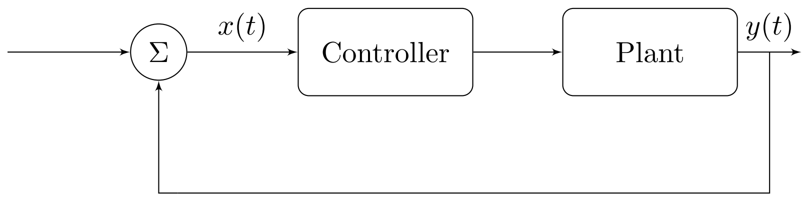


Figure 6: Example of simple control loop consisting of a controller and a plant.

3.1 Stability

In this thesis we assume that every agent within a network can be described by scalar state dynamics, such that an agent with incoming connection can be described by (2.13). If we then consider an interconnected system, Σ, with N scalar agents, where the edges between agents are non-negative, $m_{ij} > 0$ if the edge (i, j) exists and zero otherwise, the Laplacian of the graph formed by this system would be of the form

$$L(\Sigma) = \begin{bmatrix} \sum_{j\in\mathcal{N}_1^{\text{in}}} m_{1j} & -m_{12} & -m_{13} & \cdots \\ -m_{21} & \sum_{j\in\mathcal{N}_2^{\text{in}}} m_{2j} & -m_{23} & \cdots \\ -m_{31} & -m_{32} & \sum_{j\in\mathcal{N}_3^{\text{in}}} m_{3j} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}. \tag{3.1}$$

Now, we know that if the system Σ is strongly connected, the eigenvalues of $L(\Sigma)$ will be such that $0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_N$, [17]. If we now consider the situation where the agents implements an agreement protocol, then the dynamics of this agreement protocol for system Σ will become

$$\Sigma : \dot{x} = -L(\Sigma)x, \tag{3.2}$$

which will be a stable system, in the sense that the trajectory from any initial state x_0 will reach some equilibrium[3] [17]. However, if we allow for some self-loops, positive or negative, such that the diagonal of the system matrix is not necessarily equal to the sum of incoming edges for all rows, but instead some positive diagonal $A = \text{diag}(a_1, \dots, a_N)$, we can instead form the system dynamics as

$$\Sigma : \dot{x} = -(A - M)x, \tag{3.3}$$

where $(M)_{ij} = m_{ij}$. Now, if we recall the conditions for a system to be asymptotically stable, we see that the system Σ will be asymptotically stable if the real part of all eigenvalues of $-(A - M)$ are negative. Thus, if this holds true, the system Σ will converge to an equilibrium at the origin, regardless of initial state. Due to the structure of the system, we can see that this will be equivalent to $(A - M)$ being a non-singular M-matrix [14]. If we would allow for some central entity with knowledge of the entire system, the stability could be easily verified. However, through the use of local knowledge[4], this may pose to be more difficult. Nonetheless, there exists some conditions that, if each agent can set its control parameter, a_i, freely, can be verified locally and would guarantee stability of the overall system. For instance if all agents control parameter is larger than either the sum of its incoming or outgoing edges, such that the matrix $(A - M)$ is either strictly row or column diagonally dominant, this would be equivalent to $(A - M)$ being a non-singular M-matrix.

[3]Which equilibrium is dependent of the initial state as the system will converge to a weighted average of its initial state depending on connectivity.

[4]With local knowledge we mean the knowledge that some agent i has access to. If the system is directed this would include the ith row of the matrix $-(A - M)$, while if the system was undirected this would include the ith row and column.

3.2 Constraints on systems

In real-world applications it is often that actuators, or energy requirements, enforce constraints on the choices of self-feedback parameters. For instance this could be due to a finite availability of power, which would put an upper bound on the choice of self-feedback, or a lower bound on self-feedback due to dead-space in the actuators or to maintain stability, regardless of other agents choices. As we will discuss choices of self-feedback parameters throughout the following sections, we need to define what choices are feasible, in the meaning that they are not above an upper bound or below a lower bound for any given agent, when the constraints are imposed. We therefore define the set $\mathcal{A}$, which is the set of all feasible choices of self-feedback for all agents in a given system, such that the system maintains stability and does not exceed other constraints

$$\mathcal{A} = \left\{ a \in \mathbb{R}^N | (A - M) \in \mathbb{M}, A = \text{diag}(a), \underline{a}_i \leq a_i \leq \bar{a}_i, \forall i \right\}. \tag{3.4}$$

Additionally, we will assume that for all agents, the self-feedback a_i can be chosen as any value between $\underline{a}_i$ and $\bar{a}_i$, such that $\mathcal{A}$ will be a convex set.

An interesting question that arises, is how low the self-feedback can be chosen. In general the choices of self-feedback parameters an agent i can choose depend heavily on the choices of its incoming, and outgoing, neighbours. If we assume that all other agents in the network have chosen their self-feedback parameter $a_j = \bar{a}_j$, then agent i could very well choose a self-feedback, while still ensuring stability, that would be much lower than if some incoming and outgoing neighbours had chosen $a_j = \underline{a}_j$. Because of this, and the difficulties of knowing the lowest stable self-feedback through local knowledge, we will assume that $\underline{a}_i$ is set such that if all agents adapt to $\underline{a}_i$ then $(\underline{A} - M)$ would still be a non-singular M-matrix, this could for instance be achieved by setting $\underline{a}_i > \rho(M)$, for all agents.

3.3 Stability in switching systems

Even though it may be fairly simple to conclude if a time-invariant system is asymptotically stable, ensuring that a time-variant, or switching system, has the same property, proves harder. In the same sense that we can stabilise or destabilise a stable or unstable system through interconnections with some other system, the same thing can occur in a switching system. By switching system we mean that through some switching signal $\sigma(t)$ we can move between two or more systems. Switching between two systems could for instance be adding or removing an edge, or changing some agents control parameter. However, in [21] it is stated that given a family of linear systems

$$\dot{x} = A_p x,$$

such that all are asymptotically stable, $\forall p \in \mathcal{P}$, where $\mathcal{P}$ is the set of linear systems. Then, if this family of systems share a common quadratic Lyapunov function, the switching between these systems will be asymptotically stable, for any switching signal. Further [21] states that if there exists two symmetric positive definite matrices, P and Q, such that

$$A_p^T P + P A_p \leq -Q, \tag{3.5}$$

then there exists two positive constants, c and μ, such that for any switching signal and initial state, x_0, we have

$$||x(t)|| \leq c e^{-\mu t} ||x_0||, \forall t \geq 0, \tag{3.6}$$

which shows that the switching system will be asymptotically stable.

We can now show that under some circumstances on systems, any switching signal will ensure that the switching system is asymptotically stable.

Theorem 1. *Assume that a switching system switches between two systems with system matrices $(A_1 - M)$ and $(A_2 - M)$, where A_i is a positive diagonal matrix, within the set $\mathcal{A}$, and $(M)_{ij} \geq 0$ for $i \neq j$ and $(M)_{ii} = 0$, such that both $(A_1 - M)$ and $(A_2 - M)$ are non-singular M-matrices. Then any switching signal $\sigma(t)$ will ensure that the system*

$$\dot{x} = -(A_\sigma - M)x \tag{3.7}$$

will be asymptotically stable.

Proof. By [14], we have that for any M-matrix B there exists a positive diagonal matrix D, such that

$$B^T D + DB = V, \tag{3.8}$$

where V is a symmetric positive definite matrix. From (3.5), we know that if we can find two symmetric positive definite matrices P and Q, such that if for all $p \in \mathcal{P}$ and $(A_p - M)$ the following is fulfilled,

$$(A_p - M)^T P + P(A_p - M)^T \leq -Q, \tag{3.9}$$

then any switching signal will make the switching system asymptotically stable. Since both A_1 and A_2 are within $\mathcal{A}$, we know that $\underline{A} \leq A_1, A_2$, and as $(\underline{A} - M)$ will also be a M-matrix we know that there exists some matrix positive diagonal matrix D, such that

$$(\underline{A} - M)^T D + D(\underline{A} - M) = 2\underline{A}D - (M^T D + DM) \tag{3.10}$$

is positive definite. We also know that since $\underline{A} \leq A_p$, the same diagonal matrix D, can be used to form a positive definite matrix as

$$(A_p - M)^T D + D(A_p - M) = 2A_p D - (M^T D + DM), \tag{3.11}$$

since $2A_p D \geq 2\underline{A}D$. Thus, there exists a positive definite matrices $P = D$ and $Q = 2\underline{A}P - (M^T P + PM)$ such that (3.5) is fulfilled. □

4 γ-robustness

In [10], aspects of robustness in large-scale interconnected systems with respect to state deviations due to external disturbances are studied. The paper introduces the new notion of γ-robustness which is a performance measure of the ability of interconnected systems to attenuate the deviations in states stemming from external disturbances. Further, the authors continue to study how structural changes to interconnected systems can be analysed within this new notion and specifically, how the robustness properties of a system can remain unchanged during these structural changes.

4.1 Definition of γ-robustness

In order to analyse and define this measure, the large-scale interconnected systems are assumed to be comprised of N scalar subsystems of the form

$$\Sigma_i : \dot{x}_i = -a_i x_i + \sum_{j \in \mathcal{N}_i^{\text{in}}} m_{ij} x_j + d_i, \tag{4.1}$$

with both state and disturbance $x_i, d_i \in \mathbb{R}$. Further, the self-feedback a_i is assumed to be positive for all agents i, and the interconnection from agent j to agent i is assumed to be positive if and only if there exists a connection from j to i, i.e. $a_i > 0$ and $m_{ij} > 0$. This means that the interconnected system can be described as

$$\Sigma : \dot{x} = -(A - M)x + d. \tag{4.2}$$

with $A = \text{diag}(a_1, a_2, \dots, a_N) \in \mathbb{R}^{N\times N}$ and $M \in \mathbb{R}^{N\times N}$ with elements $(M)_{ij} = m_{ij}$ if j is a (in-)neighbour of i and $(M)_{ij} = 0$ otherwise, and $(M)_{ii} = 0\ \forall i$. This structure means that the system is asymptotically stable, if and only if the system matrix $(A - M)$ is a non-singular M-matrix. However, this necessary and sufficient condition on $(A - M)$ does not indicate any other performance than asymptotic stability. The authors of [10] therefore defines γ-robustness as follows.

Definition 2. *A system* (4.2) *is γ-robust (with $\gamma > 0$) if it is asymptotically stable and*

$$\max_i |x_i(t)| \leq \gamma \max_i ||d_i||_\infty \tag{4.3}$$

for all $t \geq 0$ and all trajectories satisfying $x_i(0) = 0$.

From the definition of γ-robustness it may be hard to verify if a system is indeed γ robust. However, the authors provide the following necessary and sufficient condition for γ-robustness.

Lemma 2. *A system Σ is γ-robust if and only if there exists a vector $v \in \mathbb{R}^N$ such that $v > 0$ and*

$$-(A - M)v + \mathbb{1} \leq 0,\ v \leq \gamma\mathbb{1}. \tag{4.4}$$

The proof can be found in [10], but we will restate parts of it. As the system Σ is stable and $-(A - M)v < 0$, $-(A - M)$ will be a Hurwitz matrix, which gives that the real part of all eigenvalues of $-(A - M)$ will be negative, and due to the structure of $-(A - M)$ this is equivalent of saying that $(A - M)$ is a non-singular M-matrix. Then, with an input $d(\cdot)$ such that $\max_i ||d_i||_\infty \leq 1$, the following will be satisfied

$$-(A - M)v \leq -\mathbb{1} \leq d(t) \leq \mathbb{1} \leq (A - M)v, \tag{4.5}$$

for all t.

The smallest γ for which a system is γ-robust can the be found through (4.5). Since $(A-M)$ is a non-singular M-matrix for all asymptotically stable systems, the smallest v for which the system is γ-robust will be

$$u = (A - M)^{-1}\mathbb{1}, \tag{4.6}$$

and the lowest γ for which a system, (4.2), is γ-robust will be $\gamma = \max_i u_i$ [10].

4.2 Finding the vector u

As we can see in (4.6) it is possible to find the vector u by knowing the matrix $(A - M)$. In general, this is not local knowledge, but through the use of distributed estimation each agent in the system can gain knowledge of the vector u. For instance, the algorithm presented in [22] can be used to find the vector u in a distributed and asynchronous fashion. The algorithm presented can be used to solve linear equation systems where each agent has access to a unique subset of the equation system, A_i and b_i, such that the equation system $Ax = b$ can be written as

$$A = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_M \end{bmatrix}_{n\times n}, b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_M \end{bmatrix}_{n\times 1}. \tag{4.7}$$

At each iteration, of the synchronized version of the algorithm[5], each agent updates its estimate as the orthogonal projection on the kernel of A_i of the mean of its and its neighbours estimate, and then adds a correction term based on its subset b_i. This means that the trajectory of each agents estimate is governed by

$$x_i[k+1] = \frac{1}{|\bar{\mathcal{N}_i}|} P_i \left(\sum_{j \in \bar{\mathcal{N}_i}} x_j[k] + x_i[k] \right) + A_i^T (A_i A_i^T)^{-1} b_i, \; k \geq 1. \tag{4.8}$$

Further, the authors of [22] continue to prove that this converges to the solution from any initial state if the following conditions are fulfilled, there exists a unique solution to $Ax = b$, each subset of the algorithm has full row rank and the graph is strongly connected. In order to find the vector u the system that is solved by the algorithm will be $(A - M)u = \mathbb{1}$, and each agent will have access to the local knowledge $(A - M)_{i\bullet}$, which will have full row rank, further, the other conditions posed in [22] can be fulfilled by the system Σ, with the following assumptions. First, if Σ is asymptotically stable, $(A - M)$ will be a non-singular M-matrix and there will exist a unique solution. Second, if Σ is strongly connected then the second condition is fulfilled. However, if Σ is only weakly connected the choice of setting $\bar{\mathcal{N}_i} = \mathcal{N}_i^{\text{in}} \cup \mathcal{N}_i^{\text{out}}$, i.e. each agents shares its estimate with not only its outgoing neighbours but also its incoming neighbours, then the resulting communication graph will be connected, such that there exists a path between every pair of agents, and the second condition will be fulfilled.

Remark 3. *This choice of $\bar{\mathcal{N}_i}$ might not make sense for all networks and applications, this means that if the application or network is such that the agents cannot share its estimate with its incoming neighbours, then the resulting directed network needs to be strongly connected.*

4.3 The bound u as a function of a [6]

To further extend the existing results regarding γ-robustness we will investigate how the bound u behaves as a function of the self-feedback parameters A. Throughout this section we will use the vector a (*without* subscript) as the column vector of *all* self-feedback parameters.

We start with the following Lemma, which shows that (4.6) as a function of a_i is elementwise monotonic and non-increasing, such that decreasing any control parameter a_i leads to an elementwise increase in u.

Lemma 4. *Let u be defined as in (4.6), then u as a function of a_i, i.e. $u(a_i) = (A - M)^{-1}\mathbb{1}$ with $[A]_{ii} = a_i$, will be non-increasing in the set $\mathcal{A}$. Such that, if $a_i(1) \leq a_i(2)$, $a_i(1), a_i(2) \in \mathcal{A}$, then $u(a_i(2)) \leq u(a_i(1))$ elementwise. Additionally, this non-increasing function will be continuous for all $a \in \mathcal{A}$.*

Proof. Note that $(A - M)^{-1}$ can be rewritten as

$$(A - M)^{-1} = (\mathbb{I} - A^{-1}M)^{-1}A^{-1}, \tag{4.9}$$

and with $(A - M)$ being a non-singular M-matrix, then

$$(\mathbb{I} - A^{-1}M)^{-1} = \mathbb{I} + \sum_{k=1}^{\infty} (A^{-1}M)^k, \tag{4.10}$$

[5]We have chosen to use the synchronized version of the algorithm in order to shorten explanations, as we are only interested in showing that it is a reasonable assumption to make that each agent can know its respective element in u through a decentralized algorithm.

[6]Some of the results presented in this section have previously been submitted as part of a paper, see [23], we will not cite this paper further, in order to increase readability of the section.

see [14]. Now, let $A_1 = \text{diag}(a_1, \dots, a_i(1), \dots, a_N)$ and $A_2 = \text{diag}(a_1, \dots, a_i(2), \dots, a_N)$, which implies $(A_1 - M)$ and $(A_2 - M)$ are both non-singular M-matrices and $A_2^{-1} \le A_1^{-1}$. Then by (4.9)-(4.10) $u(a_i(2)) \le u(a_i(1))$, such that $u(a_i)$ is elementwise non-increasing. □

Lemma 4 shows that, regardless of system structure, u as a function of a single agents self-feedback is non-increasing. Additionally, this means that the direction in which all bounds move when altering any self-feedback parameter can be determined through local knowledge only. It also shows that any infinitesimal change in a_i results in an infinitesimal change in the elements of u, showing that $u(a_i)$ is continuous. Further, we can continue with the following Lemma to show that given a stable system, and the set $\mathcal{A}$, u_i as a function of a is convex.

Lemma 5. *Given a stable system Σ and a constraint set $\mathcal{A}$, such that all systems formed by* $\text{diag}(A) = a$ *with $a \in \mathcal{A}$ are stable, and let u_i be the ith element of the vector defined as in* (4.6)*. Then u_i as a function of a, i.e. $u_i(a) = e_i^T(A-M)^{-1}\mathbb{1}$ with* $\text{diag}(A) = a$*, will be convex on the set $a \in \mathcal{A}$.*

To prove Lemma 5 we start with repeating that a sufficient condition for a function to be convex over a domain is that its Hessian is positive semi-definite over said domain, the proof will therefore be showing that the Hessian is positive semi-definite. We start with the partial derivative of u_i w.r.t. a_j

$$\frac{\partial u_i}{\partial a_j} = \frac{\partial e_i^T(A-M)^{-1}\mathbb{1}}{\partial a_j} = \underbrace{-e_i^T(A-M)^{-1}e_j}_{(A-M)^{-1}_{ij}} \underbrace{e_j^T(A-M)^{-1}\mathbb{1}}_{u_j}. \tag{4.11}$$

Which is due to the following property of the derivative of a matrix inverse

$$\frac{\partial A^{-1}}{\partial x} = -A^{-1}\frac{\partial A}{\partial x}A^{-1}$$

together with

$$\frac{\partial (A-M)}{\partial a_j} = e_j e_j^T.$$

As we continue the equations will become longer and more complex, therefore we introduce the following notation

$$U = \begin{bmatrix} u_1 & & 0 \\ & \ddots & \\ 0 & & u_N \end{bmatrix} \implies u = U\mathbb{1} \tag{4.12}$$

in order to shorten our expressions and increase the clarity. This means that the partial derivative of u_i w.r.t. a will be

$$\frac{\partial u_i}{\partial a} = -\left(e_i^T(A-M)^{-1}\right)U. \tag{4.13}$$

We can now continue with our build-up for the proof and show the individual elements of the Hessian as

$$\begin{aligned}\frac{\partial^2 u_i}{\partial a_j \partial a_k} &= e_i^T(A-M)^{-1}e_k e_k^T(A-M)^{-1}e_j e_j^T(A-M)^{-1}\mathbb{1} \\ &+ e_i^T(A-M)^{-1}e_j e_j^T(A-M)^{-1}e_k e_k^T(A-M)^{-1}\mathbb{1} \\ &= (A-M)^{-1}_{ik}(A-M)^{-1}_{kj}u_j + (A-M)^{-1}_{ij}(A-M)^{-1}_{jk}u_k. \end{aligned} \tag{4.14}$$

The Hessian of $u_i(a)$ can then be expressed as

$$\frac{\partial^2 u_i}{\partial a^2} = \left(e_i^T(A-M)^{-1} \otimes \mathbb{1}\right) \odot U(A-M)^{-T} + \left(e_i^T(A-M)^{-1} \otimes \mathbb{1}\right)^T \odot (A-M)^{-1}U \tag{4.15}$$

where $\otimes$ is the Kronecker product and $\odot$ is the Hadamard product. We are finally ready to show that the Hessian is positive semi-definite and $u_i(a)$ is a convex function.

Proof. A matrix is positive semi-definite if it is a Hermitian matrix with positive eigenvalues. Which according to Sylvester's criterion is equivalent to saying if and only if it is Hermitian with all principal minors non-negative. First, (4.15) is Hermitian since

$$\begin{aligned}&\left(e_i^T(A-M)^{-1} \otimes \mathbb{1}\right) \odot U(A-M)^{-T} + \left(e_i^T(A-M)^{-1} \otimes \mathbb{1}\right)^T \odot (A-M)^{-1}U = \\ &\left(e_i^T(A-M)^{-1} \otimes \mathbb{1}\right) \odot U(A-M)^{-T} + \left(\left(e_i^T(A-M)^{-1} \otimes \mathbb{1}\right) \odot U(A-M)^{-T}\right)^T,\end{aligned}$$

and we can continue showing that it will have all principal minors non-negative. We start with the second matrix factor

$$U(A-M)^{-T}. \tag{4.16}$$

Which will have all principal minors positive, since $(A-M)^{-T}$ is an inverse M-matrix, having all principal minors positive, [15], and U being a positive diagonal matrix, also having all principal minors positive. Continuing, the first factor, $\left(e_i^T(A-M)^{-1} \otimes \mathbb{1}\right)$, will be a rank one matrix with all non-negative elements. Now, the Hadamard product of these two matrices, $\left(e_i^T(A-M)^{-1} \otimes \mathbb{1}\right) \odot U(A-M)^{-T}$, can be interpreted as a scaling of the linearly independent columns in $U(A-M)^{-T}$. This gives the principal minors, with indices $I = J$,

$$\det\left(\left(e_i^T(A-M)^{-1} \otimes \mathbb{1}\right) \odot U(A-M)^{-T}\right)_{[I,J]} = \prod_{j \in J}(A-M)^{-1}_{ij} \det\left(U(A-M)^{-T}\right)_{[I,J]}. \tag{4.17}$$

Hence, all principal minors in (4.17) will be non-negative and (4.15) will be a Hermitian matrix with all principal minors non-negative. Therefore, the Hessian, according to Sylvester's criterion, is positive semi-definite and u_i, as a function of a, will be convex. □

Corollary 6. *Now as $(A-M)$ will be a M-matrix for all $a \in \mathcal{A}$, we know that $(A-M)^{-1}$ will be defined and have at least one positive element in each row. Further, u_i will be defined, and elementwise positive, for all $a \in \mathcal{A}$. Because of this $\frac{\partial u_i}{\partial a}$ in (4.13) will be continuous and non-positive, for all $a \in \mathcal{A}$. This means that $u_i(a)$ will be everywhere differentiable, for all $a \in \mathcal{A}$, thereby a continuous and smooth function.*

Now, from Lemma 5 we know that $u_i(a)$ is convex in the set $a \in \mathcal{A}$, for all i, and from Lemma 4 we know that $u(a_i)$ is non-increasing in the set $a \in \mathcal{A}$, for all i. Additionally, from Corollary 6 we know that $u_i(a)$ is continuous and smooth for all $a \in \mathcal{A}$. This knowledge can be used when pursuing some alterations of algorithms that will be presented later in the thesis. These results will therefore only be used in the discussion in Section 6.

Given the convex set $\mathcal{A}$, defined as in (3.4), it is interesting to investigate what set of feasible vectors u is formed. With this we mean all possible vectors u, given any possible combination of control parameters within $\mathcal{A}$. Using the knowledge gained from Lemma 4 and the convex set $\mathcal{A}$ we can show that the set of feasible vectors u will be convex.

Theorem 7. *Let the set $\mathcal{A}$ be defined as a convex set of (box-)constraints, such that any combination of control parameters in $\mathcal{A}$ results in a stable system, such as* (3.4)*. Then the set of vectors resulting from all combinations of control parameters, $\mathcal{U}$, defined as*

$$\mathcal{U} = \left\{u = (A - M)^{-1}\mathbb{1} \in \mathbb{R}^N, A = \operatorname{diag}(a) | a \in \mathcal{A}\right\}, \tag{4.18}$$

will be a convex set.

Proof. First, by Corollary 6 $u_i(a)$ is continuous for all $a \in \mathcal{A}$, which gives that the set $\mathcal{U}$ is connected. Further, since the boundary of $\mathcal{A}$ is included, the boundary of $\mathcal{U}$ is included, which gives that $\mathcal{U}$ is a connected and compact set. We will come back to, and use this property of $\mathcal{U}$ later in the proof. Next, we know that a set is convex if and only if any convex combination of two elements in the set is also included in the set. Thus, if $\mathcal{U}$ is a convex set then

$$(1 - \lambda)u(A_1) + \lambda u(A_2) \in \mathcal{U}, \lambda \in [0, 1] \tag{4.19}$$

has to be true for all $u(A_1), u(A_2) \in \mathcal{U}$. Now, we know that $A_1, A_2 \in \mathcal{A}$, from the definition of $\mathcal{U}$ in (4.18), we also know that $0 \leq u(A_1), u(A_2) < \infty$. Before we show that (4.19) is true for all $u(A)$ and λ, we start with showing that segments between points such that $A_1 \leq A_2$, which with Lemma 4 gives $u(A_2) \leq u(A_1)$, are in the set $\mathcal{U}$. We start with rewriting (4.19), with the assumption that $A_1 \leq A_2$, into

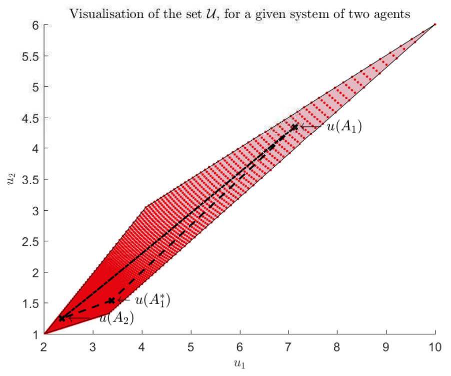
$$u(A_2) \leq (1 - \lambda)u(A_1) + \lambda u(A_2) \leq u(A_1). \tag{4.20}$$

We can also rewrite this in an elementwise inequality as

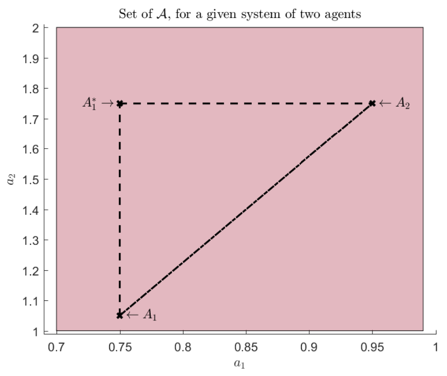
$$u_i(A_2) \leq (1 - \lambda)u_i(A_1) + \lambda u_i(A_2) \leq u_i(A_1) \tag{4.21}$$

and since $u_i(a)$ is continuous for all $a \in \mathcal{A}$, the segment $(1 - \lambda)u_i(A_1) + \lambda u_i(A_2)$ will lie in $\mathcal{U}$. We can also see that the same argument will hold if $A_2 \leq A_1$, but with reversed inequalities. Hence, segments between points $u(A_1)$ and $u(A_2)$, where either $A_1 \leq A_2$ or $A_2 \leq A_1$, lie within $\mathcal{U}$. However, we need to show that (4.19) also holds when these inequalities do not hold as well. We therefore, without loss of generality, assume that for some agent i we have $[A_1]_{ii} < [A_2]_{ii}$ while for some other agent j we have $[A_1]_{jj} > [A_2]_{jj}$ and $[A_1]_{ll} = [A_2]_{ll}$ for all other elements $l \neq i, j$. We also introduce two midpoints A_1^* and A_2^*, where $A_1^* = A_1$ for all elements except $[A_1^*]_{jj} = [A_2]_{jj}$, such that $A_1^* \leq A_2$ and $A_1^* \leq A_1$, and $A_2^* = A_2$ for all elements except $[A_2^*]_{ii} = [A_1]_{ii}$, such that $A_2 \leq A_2^*$ and $A_1 \leq A_2^*$, in a more general setting the number of midpoints could be expanded in a similar fashion.
By (4.20)-(4.21) we can note that all four segments, $[u(A_1), u(A_1^*)]$, $[u(A_1^*), u(A_2)]$, $[u(A_1), u(A_2^*)]$ and $[u(A_2^*), u(A_2)]$, are included in $\mathcal{U}$. Now, since $u_i(a)$ is continuous for all $a \in \mathcal{A}$ and i, by Corollary 6, and $\mathcal{U}$ is connected and compact, the segment $[u(A_1), u(A_2)]$ must also be part of $\mathcal{U}$. If it was not, then $u_i(a)$ would not be continuous for some point A^* between A_1 and A_2 which would mean that (4.13) would not be defined at A^*. But since (4.13) is defined for all $A \in \mathcal{A}$, since for all $A \in \mathcal{A}$ we have $(A - M)$ will be a non-singular M-matrix, this would mean that $A^* \notin \mathcal{A}$ and that $\mathcal{A}$ is not a convex set. Thus, for a given convex set $\mathcal{A}$, defined as in (3.4), the set $\mathcal{U}$, defined as in (4.18), is convex. □

By utilizing that this set $\mathcal{U}$ is convex we will be able to show that standard techniques could be used in a centralised controller to form and control the network performance. Which will motivate the decentralised algorithm that will be presented in section 5.5.



(a) Visualisation of how the set $\mathcal{U}$ would appear given a system and set $\mathcal{A}$ according to (4.22), together with the segments between the two points $u(A_1)$ and $u(A_2)$.



(b) Visualisation of the set $\mathcal{A}$, in shading, and the segments between the points A_1 and A_2.

Figure 7: Plots of the two sets $\mathcal{A}$ and $\mathcal{U}$ together with the connection between the segments in $\mathcal{A}$ and $\mathcal{U}$.

As it may be difficult to understand how the set $\mathcal{U}$ looks, a visualisation of the set was made for a two agent system. This system was defined as

$$(A - M) = \begin{bmatrix} a_1 & -1 \\ -0.5 & a_2 \end{bmatrix}, \ \mathcal{A} = \left\{ \begin{array}{c} 0.7 \leq a_1 \leq 0.99 \\ 1 \leq a_2 \leq 2 \end{array} \right\}. \tag{4.22}$$

The corresponding set $\mathcal{U}$ can be seen in Fig. 7, the red dots indicate a sample of values of u, the shading represents the entire set $\mathcal{U}$. Additionally, we have included different segments in the sets $\mathcal{A}$ and $\mathcal{U}$, between the points A_1, A_2 and $u(A_1), u(A_2)$, to visualise the relationship between the sets.

5 Fairness [7]

5.1 Motivation

Given a network and an interconnection structure captured in M, different choices of control parameter A, i.e., different a_i, may yield the same bound γ. Such that given one system realisation, agent i could have a larger control parameter, than it would in some other, and vice versa for some other agent, but both realisations still achieving the same system wide performance measure γ. Since a greater value of a_i will yield a greater control cost, or energy, required by agent i to control its state x_i, the agents might seek to reduce a_i, in order to reduce its total energy spending. In an unconnected system, this would be a trade-off only for agent i, as a decrease in a_i would lead to an increase in susceptibility to external disturbances. However, in an interconnected system this trade-off also includes other agents, such that suitable choices of a_i ensuring the desired γ can in general not be selected independently, due to the interconnection structure in M. According to (4.6),

$$u_i = \frac{\sum_{j \in \mathcal{N}_i^{\text{in}}} m_{ij} u_j + 1}{a_i}.$$

[7]Some of the results presented in this section has previously been submitted as part of a paper, see [23]. We will not cite this paper further, in order to increase readability of the section.

In turn, in case agent i is also an in-neighbour of agent j, u_j will be a function of u_i and hence be affected bu the choice of a_i. This may then further propagate through the network. Hence, a decrease in control effort by one agent may require the increase of control costs for another agent in order for the system to remain γ-robust. Similarly, if an agent increases its control effort, in order to reduce u_i, some connected agent could also see a reduction in their respective element of u, such that their control effort would be unnecessarily high.

In turn, in case all agents choose the same control parameter a_i, the resulting u_i may indeed be very unevenly distributed as some agent may simply have a larger amount of incoming edges, leading in general to larger u_i. In contrast, despite equal control efforts, other agents may achieve a very low value of u_i, which could be considered as an unnecessary expense in control costs in case there exists a large margin between u_i and γ. This means that throughout the system, the distribution of control effort and resulting u_i may be distributed very unevenly, or unfairly. Below, we will consider the question on how overall performance criteria of a system Σ can be guaranteed with a *fair* distribution of local parameters.

5.2 Formal definition of fairness

Formally, we define the general notion of *fairness* according to a quantity of choice χ as follows:

Definition 3. *A system is (globally) χ-fair, for a property χ, if all agents in the system share the same size of the given property, such that $\chi_i = \chi$ for all agents.*

As we consider large-scale interconnected systems and seek decentralised control solutions that do not require centralised information, χ-fairness as a global property cannot be easily verified by agents with local knowledge only. Therefore, we will define local χ-fairness as follows.

Definition 4. *An agent i with its set of neighbours, $\bar{\mathcal{N}_i}$, is locally χ-fair if $\chi_i = \chi_j \, \forall j \in \bar{\mathcal{N}_i}$.*

Note that, the set of neighbours $\bar{\mathcal{N}_i}$ may be $\mathcal{N}_i^{\text{in}}$, $\mathcal{N}_i^{\text{out}}$, $\mathcal{N}_i^{\text{in}} \cup \mathcal{N}_i^{\text{out}}$ or in fact any other set of local neighbours with whom agent i compares χ_i for the sake of determining χ-fairness. However, depending on connectivity and application of a system, some choices of $\bar{\mathcal{N}_i}$ might not be possible or might be unnecessary, as no obvious benefits will emerge.

Lemma 8. *Assume that a system Σ contains a spanning rooted out-branching graph, and that it is locally χ-fair according to the set of in-neighbours, i.e., $\bar{\mathcal{N}_i} = \mathcal{N}_i^{\text{in}}$. Then $\chi_i = \chi_j$, $\forall i \neq j$, and $\chi_i = \chi_{j \in \mathcal{N}_i^{\text{in}}}$, $\forall i$. Further, if $\chi_i = \chi_{j \in \mathcal{N}_i^{\text{in}}}$, $\forall i$, then $\chi_i = \chi_j$,$\forall i \neq j$, and the system is χ-fair.*

Proof. Since the underlying graph contains a spanning rooted out-branching there exists a vertex, denoted r, such that there exists a directed path from r to all other vertices. Then, $\chi_r = \chi_{i \in \mathcal{N}_r^{\text{out}}} = \Omega^{\chi}_{j \in \mathcal{N}_i^{\text{out}}} = \cdots = \chi_N$ which that $\chi_i = \chi_j \forall i \neq j$. □

By [17] any strongly connected graph contains a spanning rooted out-branching, such that $\bar{\mathcal{N}_i} = \mathcal{N}_i^{\text{in}}$ would be a suitable choice to ensure that the system is χ-fair if and only if it is locally χ-fair. Equivalently, if the direction of the edges of the graph are reversed, then one could choose to use $\bar{\mathcal{N}_i} = \mathcal{N}_i^{\text{out}}$ and the above Lemma would instead hold for the new graph and choice of $\bar{\mathcal{N}_i}$. However, if the system is instead only weakly connected, i.e. its undirected graph is connected, then the choice $\bar{\mathcal{N}_i} = \{\mathcal{N}_i^{\text{in}}, \mathcal{N}^{\text{out}}\}$ will ensure that local fairness for all neighbours is equivalent as global fairness.

5.3 Formal definition of unfairness

In case systems cannot reach fairness due to limitations or restrictions on the choice of parameters, such as in Section 3.2, the notion of fairness may offer little insight as a system may only be either fair or not. Then, a measure of distance between the current parameter setting and a setting that would reach a fair system may be used to quantify the unfairness. We therefore define a measurement of global unfairness for an arbitrary local parameter χ as

$$\Omega^{\chi} = \sum_{e\in\mathcal{E}} (\chi_{\text{in}(e)} - \chi_{\text{out}(e)})^2, \tag{5.1}$$

where the indices in(e) and out(e) refer to the nodes connected by edge e. In a weakly connected network, Ω^{χ} will only be equal to zero when all agents share the same value of χ, i.e. the system is χ-fair. However, as this measurement considers all edges in the network, it can only be measured by some entity with global knowledge, which means that only networks with some central controlling entity or networks that have edges between all pairs of agents can measure global unfairness. Therefore, a measurement of unfairness which can be measured locally will be defined as

$$\Omega_i^{\chi}(\bar{\mathcal{N}_i}) = \sum_{j\in\bar{\mathcal{N}_i}} (\chi_i - \chi_j)^2\,, \tag{5.2}$$

which means that $\Omega_i^{\chi} = 0$ only when $\chi_i = \chi_j$, $\forall j \in \bar{\mathcal{N}_i}$ where we consider a general set of neighbours $\bar{\mathcal{N}_i}$. If χ_i is compared to χ_j of all direct neighbours with incoming edges to i, unfairness is a function of $\bar{\mathcal{N}_i} = \mathcal{N}_i^{\text{in}}$. But other sets of neighbours, for example also considering neighbours that are connected with edges from agent i, i.e, $\bar{\mathcal{N}_i} = \mathcal{N}_i^{\text{in}} \cup \mathcal{N}_i^{\text{out}}$ may be considered for the purpose of measuring unfairness.

Depending on the choices of neighbourhood set $\bar{\mathcal{N}_i}$ in (5.2), the relation between local and global unfairness will vary. However, (5.2) is equivalent to the summation over the edges in $\mathcal{E}$ which end or start in agent i. Thereby, $\Omega^{\chi} = \sum_i \Omega_i^{\chi}(\bar{\mathcal{N}_i})$ for the choices $\bar{\mathcal{N}_i} = \mathcal{N}_i^{\text{in}}$ or $\bar{\mathcal{N}_i} = \mathcal{N}_i^{\text{out}}$, while the choice $\bar{\mathcal{N}_i} = \mathcal{N}_i^{\text{in}} \cup \mathcal{N}_i^{\text{out}}$ results in $\sum_i \Omega_i^{\chi}(\bar{\mathcal{N}_i}) = 2\Omega^{\chi}$.

Remark 9. *Different choices of χ may be appropriate in different circumstances: i.e., $\chi = u$ is a reasonable choice to achieve $u = \gamma\mathbb{1}$ so that the self-feedback parameters a_i are chosen to meet the maximal bound γ while minimising the control costs. Hence some agents may have larger a_i due to their incoming connections. In contrast, $\chi = a$ should be chosen if equal distribution of the control efforts is desired.*

5.4 u-fairness

In the section above we have presented χ-fairness and measurements of unfairness for general properties of large-scale interconnected systems. We will now continue by investigating χ-fairness and unfairness with respect to the bound of state deviations due to external disturbances, which we will call u-fairness, where the vector u is defined as in (4.6). As discussed, different choices of A can lead to the same bound γ, which may lead to unnecessary control efforts or unfair distribution of state deviation bounds. Due to this we will look in to u-fairness by starting with the properties of a u-fair system.

Lemma 10. *If the matrix $(A-M)$ is strictly diagonally row dominant, then there exists a vector $\gamma\mathbb{1}$ with $\gamma \geq \gamma_i > 0$, $\forall i$, where γ_i satisfies $a_i = \sum_{j\in\mathcal{N}_i^{\text{in}}} m_{ij} + \frac{1}{\gamma_i}$, such that $-(A-M)\gamma\mathbb{1} + \mathbb{1} \leq 0$. Further, if for a system there exists a vector of the form $\gamma\mathbb{1}$ such that $-(A-M)\gamma\mathbb{1} + \mathbb{1} \leq 0$ then the system will be stable and $(A-M)$ must be diagonally row dominant.*

Proof. If $(A-M)$ is strictly row diagonally dominant, i.e. $a_i - \sum_{j\in\mathcal{N}_i} m_{ij} > 0, \forall i \Leftrightarrow (A-M)\mathbb{1} > 0$, then there will exist a vector $\gamma\mathbb{1}$ such that $-(A-M)\gamma\mathbb{1} + \mathbb{1} \leq 0$ for some $\gamma > 0$. Further, if $-(A-M)\gamma\mathbb{1} + \mathbb{1} \leq 0$, with $\gamma \geq \gamma_i$, then $(A-M)\mathbb{1} > 0$ and $(A-M)$ is diagonally row dominant. Additionally, stability follows from $(A-M)$ being strictly diagonally row dominant which is equivalent to $(A-M)$ being a non-singular M-matrix, [14]. □

Thus, we now know how a u-fair system will be structured and we can see that it only requires local knowledge for an agent to estimate if the system can be made u-fair.

Reaching u-fairness in systems without constraints

Lemma 10 on strictly diagonally dominant matrices leads to the following corollary in the context of reaching u-fairness.

Corollary 11. *If a system is u-fair, i.e. $-(A-M)u + \mathbb{1} = 0$ with $u = \gamma\mathbb{1}$, then $(A-M)$ must be strictly diagonally row dominant with*

$$a_i = \sum_{j\in\mathcal{N}_i^{\text{in}}} m_{ij} + \frac{1}{\gamma}, \text{ for all } i, \tag{5.3}$$

with γ being known by all agents and this can be chosen arbitrarily small or large prior to system initialisation. Further, if the control parameter a_i of all sub-systems can be chosen freely, then the system can be made u-fair with bound γ by adapting all a_i accordingly.

Remark 12. *Note that the condition* (5.3) *in Corollary 11 requires, apart from the desired global bound γ, only local knowledge, i.e., the weights of incoming edges m_{ij}.*

Given a system Σ, which is not u-fair, the agents may adjust their control parameters according to (5.3) asynchronously. In general, this adaptation may lead to $A-M$ being, at least temporarily, not asymptotically stable. Hence, great care should be taken to ensure synchronous adaptation or other algorithms, that ensure asymptotic stability for all times. In case the system Σ is, however, strictly diagonally row dominant before adapting the control parameters, then adapting the control parameters in order to satisfy (5.3) may lead to temporarily violating the overall bound γ, i.e., $\max_i u_i > \gamma$, but the system will be asymptotically stable at all times during the adaptation of control parameters.

Remark 13. *However, it should be noted that any on-line adaptation of control parameters, i.e. an adaptation of control parameters as the system is running, may lead to the time-varying system becoming unstable. But as of Theorem 1 we know that if both systems, before and after switching, are asymptotically stable and only control parameters have been changed, such that $(A_b - M)$ and $(A_a - M)$ are both M-matrices, then any switching signal will ensure asymptotic stability.*

Systems with constraints

In real world applications, constraints may be imposed on the control parameters, due to limits on the achievable control costs or actuator limitations as discussed in 3.2. These constraints can be used to form a set of control parameters as in Section (3.4), such that the system always remains stable. However, in this section, we assume that the constraints are such that the system matrix can still be made strictly diagonally row dominant[8], such that

$$\bar{a}_i > \sum_{j\in\mathcal{N}_i^{\text{in}}} m_{ij} \text{ for all } i. \tag{5.4}$$

[8]For scenarios where (5.4) cannot be satisfied for all agents, see the following section.

Due to this constraint, it may not be possible to choose the control parameter a_i according to (5.3) for all i for a given, desired γ. But according to Lemma 10, it is still possible to choose suitable control parameters to construct a u-fair system.

Hence, assuming that the constraints are such that the set of self-feedback parameters is defined as in (3.4) with the upper bound set as (5.4), we intend to find a bound u^* with which the system can reach fairness. We, therefore, alter (5.3) into

$$a_i = \sum_{j \in \mathcal{N}_i^{\text{in}}} m_{ij} + \frac{1}{u^*} \tag{5.5}$$

such that the constraints are satisfied for all i while reaching u-fairness, i.e., achieving $u = u^*\mathbb{1}$, where $u^* \in U$ with U defined by the intersection of the local bounds

$$U = \cap_{i=1}^{N} U_i.^9 \tag{5.6}$$

The local bounds are restricted by the lowest, fair u that agent i can achieve if the vector u would have the form $\underline{u}_i\mathbb{1}$ with (according to (5.5))

$$\underline{u}_i = \frac{1}{\bar{a}_i - \sum_{j \in \mathcal{N}_i^{\text{in}}} m_{ij}}.$$

The upper bound $\bar{u}_i$ will be achieved when agent i reduces it self-feedback to $\underline{a}_i$, and as this lower bound on a_i is defined as the lowest value such that $(A - M)$ is still a non-singular M-matrix. This means that the vector u in (4.6) will be bounded from above, meaning that there will exist a $\bar{u}^* < \infty$ with $u < \bar{u}^*\mathbb{1}$, such that the local bounds will be

$$U_i = [\underline{u}_i, \bar{u}_i], \text{ with } \bar{u}_i < \infty. \tag{5.7}$$

Further, we can define the projection operator, on the set U_i, as

$$P_{U_i}[\nu] = \arg \min_{u_i \in U_i} \|\nu - u_i\|. \tag{5.8}$$

The following theorem shows the existence of a suitable consensus algorithm to reach u-fairness.

Theorem 14. *Consider a system Σ as in* (4.2) *with a weakly connected graph $\mathcal{G}$ and constraints on control parameters a_i as in* (3.4)*. Then, given an initial bound $u_i(0) \in U_i$ for all i, if the agents update their u_i iteratively by the following projected consensus algorithm*

$$\hat{u}_i[k+1] = P_{U_i}\left[\frac{1}{|\mathcal{N}_i| + 1}\left(\sum_{j \in \mathcal{N}_i} \hat{u}_j[k] + \hat{u}_i[k]\right)\right], \tag{5.9}$$

where $\mathcal{N}_i = \mathcal{N}_i^{\text{in}} \cup \mathcal{N}_i^{\text{out}}$, and the control parameters are hence updated by

$$a_i[k+1] = \sum_{j \in \mathcal{N}_i^{\text{in}}} m_{ij} + \frac{1}{\hat{u}_i[k+1]}, \tag{5.10}$$

for $k \geq 1$, then the system will achieve consensus in u such that $u = u^*\mathbb{1}$, i.e., the system will achieve fairness while respecting the constraints.

[9] Note that this is not the same set as the set $\mathcal{U}$ defined as in (4.18).

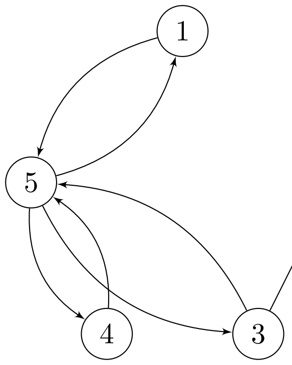


Figure 8: Graph of a initialised network for simulations.

Proof. In [24, Lem. 4] it is shown that (5.9) converges, if the set U is non-empty and convex and the following assumptions hold: If the communication edge (i, j) exists then the associated weight fulfils $w_{ij} \geq w_{ii} > 0$. Further, the weight matrix $W[k]$ formed with the weights $w_{i\bullet}$ as the ith column is doubly stochastic, the associated communication graph is strongly connected and there is a finite bound on the communication interval between agents. Additionally, in [25] it is shown that with a weight matrix $W[k]$ such that $W^T[k]$ is stochastic is sufficient for the convergence of the same algorithm. Since the undirected communication graph formed by the neighbouring agents exchanging information about their current u_i is connected (as $\mathcal{G}$ is weakly connected and we choose $\mathcal{N}_i = \mathcal{N}_i^{\text{in}} \cup \mathcal{N}_i^{\text{out}}$ in (5.9)), there exist paths between every pair of agents, the sets U_i and $U = \cap_{i=1}^N U_i$ according to (5.7) are convex and with the weights $w_{ii} = w_{ij} = \frac{1}{|\mathcal{N}_i|+1}$, the matrix $W^T[k]$ will be stochastic. Further, as we know from Section 3.3 any switch between two systems $(A_1 - M)$ and $(A_2 - M)$ will be stable as long as both $A_1, A_2 \in \mathcal{A}$. Thus the system will be stable at all times. □

Remark 15. *Note that the agents may adjust their control parameters i at discrete time instances k and communicate their current values of u_i with both their in- and out-neighbours. However, the (physical) interconnections between the agents affecting their states remain as in (4.2).*

$$(A - M) = \begin{bmatrix} 4.4715 & 0 & 0 & 0 & -0.1071 \\ 0 & 4.4715 & -2.5474 & 0 & 0 \\ 0 & 0 & 4.4715 & 0 & -2.8020 \\ 0 & 0 & 0 & 4.4715 & -2.0362 \\ -2.2732 & 0 & -2.2294 & -1.1767 & 4.4715 \end{bmatrix} \tag{5.11}$$

To show the convergence of the algorithms a simulation was made, for this simulation a network was initialised according to the graph depicted in Fig. 8. The control parameters and edge weights were initialised such that the system matrix $(A - M)$ was according to (5.11) and the constraints where set as

$$\underline{a}_i = 1.01\rho(M),\ \bar{a}_i = \max_i \left(\sum_{j \in \mathcal{N}_i^{\text{in}}} m_{ij} + \frac{2}{\gamma} \right), \tag{5.12}$$

with $\gamma = \max_i u_i$. The resulting trajectories of this algorithm can be seen in Fig. 9. It can be noted that this algorithm does not converge to the lowest possible fair u, but instead some weighted average of its initial estimate, therefore one might be tempted to seek another approach which could both reduce unfairness while also penalizing larger elements in u, such an algorithm will be discussed in the sections that follow.

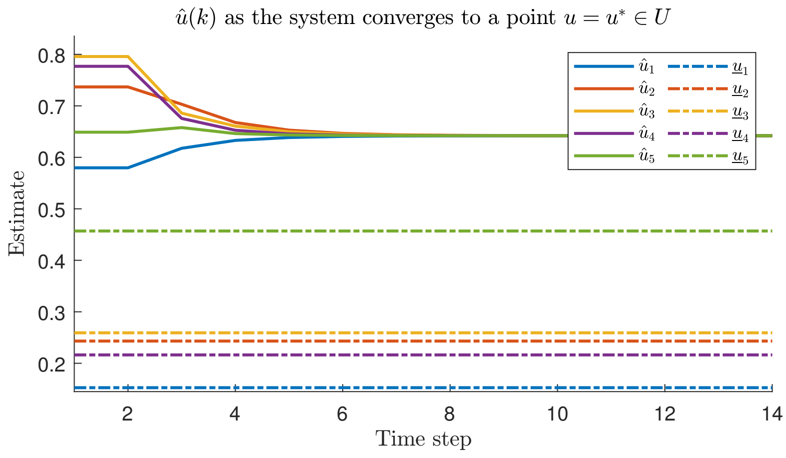


Figure 9: A network of $N = 5$ agents converging to a consensus on a fair u, this algorithm (5.9)-(5.10) ran synchronously over the network initialised as (5.11)

5.5 Reducing u-unfairness

Now, as there may be constraints on the systems, due to for instance limitations in actuators, and we cannot always assume that the constraints are such that (5.5) is fulfilled. This would mean that for a given system Σ, the constraint set $\mathcal{A}$, defined in (3.4), is such that Σ cannot become strictly diagonally dominant, which according to Lemma 10 means that it cannot become u-fair. Additionally, it could be so that the system Σ can be made fair, but at the prize of a very large bound γ. All these cases results in a situation where it may not be possible or desirable to reach u-fairness, but instead reduce u-unfairness and penalize large bounds. This can be interpreted as solving an optimisation problem with an objective function constructed as a combination of the unfairness measurement, defined in (5.1), and the penalty for large u, the combination would then, for a system Σ, be of the form

$$f(u) = \alpha \sum_{e\in\mathcal{E}} \left(u_{\text{in}(e)} - u_{\text{out}(e)}\right)^2 + \beta u^2. \tag{5.13}$$

Now, by noting that $\left(u_{\text{in}(e)} - u_{\text{out}(e)}\right)$ can be rewritten using the incidence matrix, from the graph $\mathcal{D}$ depicting the structure of Σ, we get $\left(u_{\text{in}(i)} - u_{\text{out}(i)}\right) = B_{i\bullet}u$. We can therefore rewrite the measurement of unfairness as

$$\sum_{e\in\mathcal{E}} \left(u_{\text{in}(e)} - u_{\text{out}(e)}\right)^2 = u^T B^T B u = u^T L u$$

where L is defined according to (2.2). The function $f(u)$ in (5.13) can now be rewritten as

$$f(u) = u^T(\alpha L + \beta \mathbb{I})u,$$

which will be a strictly convex function if $\beta > 0$. As said, we may have some constraints on a. These will, as of Theorem 7, be translated into the convex set $\mathcal{U}$, assuming that the set $\mathcal{A}$ is defined as in (3.4). The optimisation problem can now be stated as

$$\min_{u\in\mathcal{U}} f(u) = u^T(\alpha L + \beta \mathbb{I})u. \tag{5.14}$$

As $f(u)$ in (5.14) is a convex function and $\mathcal{U}$ is convex by Theorem 7, the projected gradient method can be used. However, the projection onto the set $\mathcal{U}$ can be very cumbersome to find, as the constraints are not stated in u, but rather in A. We therefore seek other types of constraints that could ensure $u \in \mathcal{U}$, without having to find this projection. We therefore find that we can achieve this with two sets of constraints, with the first given as

$$-(\bar{A} - M)u + 1 \leq 0. \tag{5.15}$$

Since we can rewrite $(\bar{A} - M)$ into $(\bar{A} - M) = (A - M) + A'$, where $A' = \bar{A} - A$, we get $(\bar{A} - M)(A - M)^{-1}\mathbb{1} = (A' + (A - M))(A - M)^{-1}\mathbb{1} \geq \mathbb{1}$ for all $u \in \mathcal{U}$. The second set of constraints are to limit u from growing to large and becomes,

$$(\underline{A} - M)u - 1 \leq 0, \tag{5.16}$$

which gives the combination of constraints $\underline{A} \leq A + A' \leq \bar{A}$ for all $u \in \mathcal{U}$. Thus, (5.14) instead becomes

$$\begin{aligned} \min_{u \in \mathbb{R}_+^N} f(u) &= u^T \frac{1}{2}(\alpha L(\mathcal{G}) + \beta \mathbb{I})u \qquad (5.17) \\ \text{subject to}\, g_i(u) &= -(\bar{A} - M)_{i\bullet} u + 1 \leq 0,\ i = 1, \dots, N \\ h_i(u) &= (\underline{A} - M)_{i\bullet} u - 1 \leq 0,\ i = 1, \dots, N \end{aligned}$$

with $\alpha, \beta > 0$. It is now possible to find the optimum using standard techniques, [19], if we were to solve the problem centralised. But in order to solve this distributively we see that we can write $f(u) = \sum_i f_i(u_i)$, with

$$f_i(u) = \frac{1}{2}\left(\alpha \sum_{j \in \bar{\mathcal{N}}_i} (u_i - u_j)^2 + \beta u_i^2\right). \tag{5.18}$$

If we would make the assumption that all agents not only know their own constraints, $g_i(u)$ and $h_i(u)$, but also the constraints of the entire system, or if we instead could assume that the network is fully connected, there exists algorithms that could be adapted to solve this, for instance the algorithms described in [25] and [26].

As we would like to not restrict ourselves to these assumptions we propose that each agent seeks to minimise its local objective function as a function of u_i, while still respecting the constraints in $g_i(u)$ and $h_i(u)$. This can be formulated as a projected gradient descent, such that every agent treats the estimates $\hat{u}_j$ as constants at every iteration step, with the projection being upon $\underline{a}_i \leq \frac{\sum_{j \in \bar{\mathcal{N}}_i} m_{ij}\hat{u}_j + 1}{\hat{u}_i[k]} \leq \bar{a}_i$, which gives

$$\hat{u}_i[k+1] = P_i\Big[\hat{u}_i[k] - \delta_k \frac{df_i(\hat{u}[k])}{du_i}\Big]. \tag{5.19}$$

By choosing a suitable step size, (5.19) will converge towards an optimum for the local objective function. However, it is not guaranteed that the convergence to an optimum for the local objective function ensures the convergence to an optimum for the global objective function. Additionally, it only requires local knowledge and will at every iteration lead to feasible solutions, such that if it was stopped at some time k the estimate $\hat{u}[k]$ could be used to form an asymptotically stable system.

In order to show the performance of this algorithm, a simulations was made using a network initialised as Fig. 8 and $(A - M)$ as (5.11). The constraints on control parameter where set as

$$\underline{a}_i = 1.01\rho(M),\ \bar{a}_i = 0.99 \max_i \left(\sum_{j \in \mathcal{N}_i^{\text{in}}} m_{ij}\right), \tag{5.20}$$

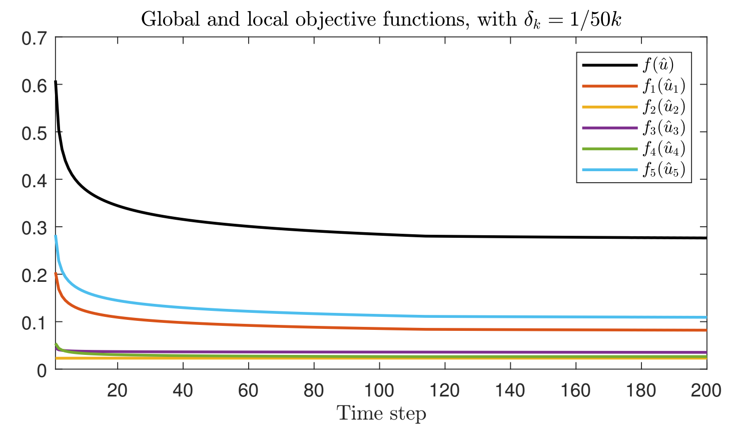


Figure 10: Trajectories of local and global objective functions as the algorithm (5.19) was run on a system initialised as (5.11) and constraints (5.20). α and β was set as $\alpha = 0.9$ and $\beta = 0.1$.

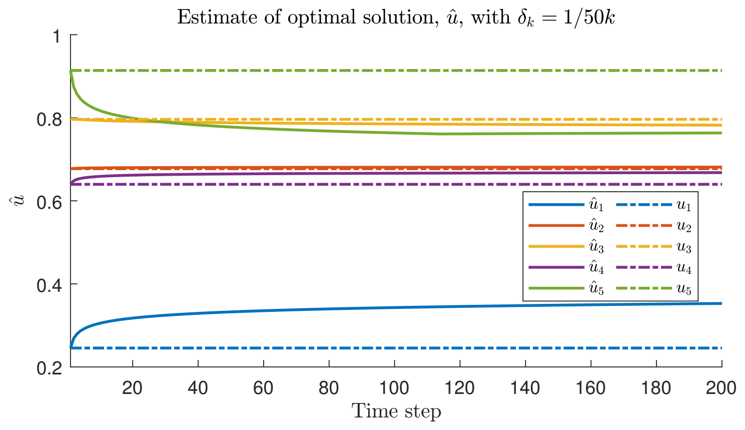


Figure 11: Trajectories of the estimates $\hat{u}^*$ as the algorithm (5.19) was run on a system initialised as (5.11) and constraints (5.20). α and β was set as $\alpha = 0.9$ and $\beta = 0.1$.

with $\gamma = \max_i u_i$. The algorithm in (5.19) ran with a step size of $\delta_k = 1/50k$ and with $\alpha = 0.9$ and $\beta = 0.1$. The resulting trajectories of local and global objective functions can be seen in Fig. 10.

In Fig. 11 the trajectories of the estimates of the optimal vector $\hat{u}^*$ can be seen. Further, we can compare the original vector u with the estimate $\hat{u}^*$ using (5.19) and a true u^* found treating the problem as a centralised optimisation problem solved using the built-in Matlab solver 'fmincon'. In Table 1, we see that both methods, (5.19) and 'fmincon', achieve a reduction in the bound γ. Using these vectors we can also see a comparison of the resulting costs of respective objective function in Table 2.

6 Discussion

We will start the discussion regarding our results, by first going through how our assumptions affect applicability and how these assumptions could be relaxed. Following this, we will discuss how model errors affect system performance and the performance of the presented algorithms. After this we will discuss the benefits of systems that have been adapted by the presented algorithms.

Table 1: Comparison of the original vector u_0 with the estimate $\hat{u}^*$, found using (5.19), and the optimum u^*, found using Matlab solver 'fmincon' on the centralised problem.

	u_0	$\hat{u}^*$	u^*
u_1	0.2455	0.3597	0.2407
u_2	0.6773	0.6819	0.3964
u_3	0.7963	0.7766	0.4405
u_4	0.6398	0.6718	0.3687
u_5	0.9138	0.7718	0.5270

Table 2: Cost of global objective function, as in (5.17) with $\alpha = 0.9$ and $\beta = 0.1$, for vectors u, $\hat{u}^*$ and u^* as in Table 1

$f(u_0)$	0.6083
$f(\hat{u}^*)$	0.2781
$f(u^*)$	0.1451

6.1 Assumptions regarding agent dynamics

Through out this thesis we have made the assumptions that all agents can be described by, essentially, the same system dynamics, which has been a linear scalar dynamic. This assumption does make it easier to formulate the presented theorems and lemmas, but also restricts the directly applicable situations. Although, with some alterations it would be possible to extend the results, presented in both this thesis and in [10], where γ-robustness was introduced, to non-scalar systems, where each agents internal state is decoupled from the other.

Consider for instance the example in Section 2.5, where agents with two states, position on x and y-axis, reach a consensus. In this example the interconnections between each agent's states are decoupled such that we can model this as two different, but identical, networks. For systems in which this is possible, we can directly apply the presented results. However, if the two networks are not identical, one would have to decide whether there could exist two different γ's, one for x-axis deviations and one for y-axis, or if the system is γ-robust for the larger of these deviation bounds. Additionally, one would have to consider the case when there is a common bound on control parameters, such that an agent cannot choose control parameter for each state freely. For instance, if there is a limit on the control parameters combined, such that $a_x + a_y \leq \bar{a}$ which could occur if there is a boundary on total available power, reducing γ of the system could then involve solving some local optimisation algorithm. Furthermore, it would also be necessary to research if an extension of the presented algorithm can be made, such that it minimizes objective function through the two, or more, local variables.

One could also consider non-scalar systems where the connections between each state is such that the state dynamics will be a M-matrix, or a $\mathbb{Z}$-matrix, if the subsystem can be stabilised by other agents. In this case, one could treat each state as its own agent and use the presented results, given that control parameters for each state can be chosen somewhat independently. For other non-scalar agents, future research could be on how to ensure γ-scalability and design decision if γ-robustness is found for agents or states. Some preliminary results regarding this can be found in the appendix of this thesis.

Further, in [27] some results regarding stability in interconnected systems with non-linear state dynamics are presented. If we assume that each agents internal system dynamic, $f_i(x)$ is some scalar valued continuous non-linear function, the system dynamics can be modelled as

$$\dot{x} = -(A - M)f(x) + d, \tag{6.1}$$

Table 3: Control parameters for the three systems during simulations of model errors.

$\tilde{A}_I$	$\tilde{A}_{\circledast}$	$\tilde{A}^*$
1.7295	1.5878	1.5966
1.7295	1.6978	2.0131
1.7295	1.9615	2.0131
1.7295	1.5970	2.0130
1.7295	1.7433	2.0131

where $f(x) = (f_1(x), \dots, f_N(x))^T$. [27] also shows some conditions on $f_i(x)$ in order for these systems to be stable. However, there could be some problems finding the lowest γ for which this system is γ robust, as this may be dependent on the state of the system.

6.2 Uncertainty simulations of γ-robustness and u-fairness

An interesting aspect to study regarding the results, is how well it behaves when there are model errors present. In order to study this we consider two different cases, the first one being that the control parameter matrix, A, contains errors and the second case being that the interconnection matrix, M, contains errors. We denote the estimated control parameter and connection matrices as

$$\tilde{A} = A + A_e, \tag{6.2}$$
$$\tilde{M} = M + M_e. \tag{6.3}$$

In the simulations we will also assume that there is only error stemming from one source, i.e. either (6.2) or (6.3), this assumption is made in order to be able to separate the effects from each of the types of errors.

Error simulations

The effect of the errors was simulated on three systems, the original initial system with control parameters $\tilde{A}_I$, the system after adaptation through the algorithm in Section 5.5 $\tilde{A}_{\circledast}$ and a system denoted as the optimal system, $\tilde{A}^*$, for which the control parameters were found by using the Matlab built-in function 'fmincon' on the centralised problem in (5.14).

$$\tilde{M} = \begin{bmatrix} 0 & 0 & 0 & 1.1873 & 0.0453 \\ 0 & 0 & 0 & 1.3961 & 0.2329 \\ 0.1534 & 0.4115 & 0 & 0 & 1.4685 \\ 0.4143 & 0 & 0.2290 & 0 & 0.6629 \\ 0 & 0.8462 & 0.7287 & 0 & 0 \end{bmatrix} \tag{6.4}$$

Control matrix error

The errors, A_e, were assumed to be a random percentage of the control parameters, where the size of this was drawn from a uniform distribution on the open interval of $(-\sigma_e, \sigma_e)$. This meant that at each simulation the true control parameter of each agent, a_i, was

$$a_i = \tilde{a}_i + \tilde{a}_i e_i,\ e_i \in (-\sigma_e, \sigma_e), \tag{6.5}$$

which meant that the true value of u for the system was calculated as

$$u = \left((\tilde{A} - A_e) - (\tilde{M} - M_e)\right)^{-1} \mathbb{1}. \tag{6.6}$$

Table 4: Comparison of mean square error between vector u of the estimated system and true system, during simulations of control parameter errors.

A_I	$A_{\hat{*}}$	A^*
0.5397	0.6130	0.0209
1.0826	1.0363	0.0243
2.4844	1.5480	0.0537
0.9074	0.8620	0.0212
1.7576	1.3265	0.0388

Table 5: Comparison of mean square error between vector u of the estimated system and true system, during simulations of connection model errors.

A_I	$A_{\hat{*}}$	A^*
0.2011	0.2369	0.0090
0.3577	0.3511	0.0078
0.8407	0.5221	0.0175
0.3194	0.3071	0.0072
0.5922	0.4495	0.0125

The resulting simulations of the three system with interconnection matrix $\tilde{M}$ according to (6.4), and $M_e = 0$, and control parameters according to Table 3 can be seen in Fig. 12. Further, in Table 4 we can compare the mean square error between the estimates and true vector u. In this table we can see that the adapted systems, $\tilde{A} = A_{\hat{*}}$ and $\tilde{A} = A^*$, may not only reduce the mean square error but also reduce the variation of mean square error between the subsystems.

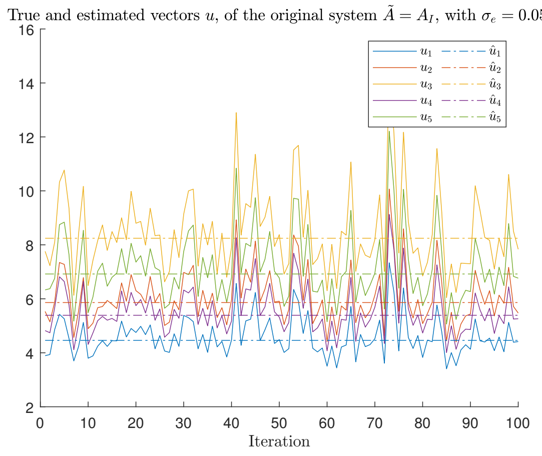
Connection matrix error

A similar test was made with only errors on the connection model, the errors on the connection model was again assumed to be a percentage of each weight and was drawn from a uniform distribution on the open interval $(-\sigma_d, \sigma_d)$. In Fig. 13 we can see some similarities between the two cases of errors, cf. Fig. 12, but it appears as the effect of errors in the edge weights cause less overall error in all three different systems.

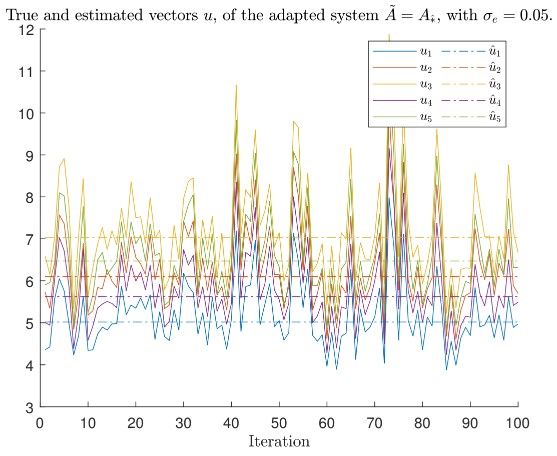
In general, it appears as if the initial systems perform worst, for most subsystems, such that the mean square error of the estimated vector u and the true u, was larger in general for the initial systems, cf. Table 4 and 5. This could be answered by the fact that the overall bound γ was reduced in all the adapted systems, or due to the fact that the adapted systems have a more fair distribution of control effort.

6.3 Benefits and disadvantages of u-fair systems

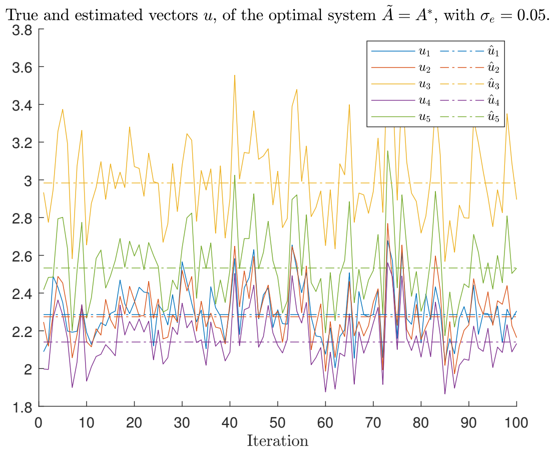
To show one of the benefits with a u-fair system and the algorithms a simulation of a system converging to a consensus was made. During these trajectories of the original, randomly initialised, system was compared to those from a system with the same interconnection matrix M, but with a control parameter matrix adapted through the algorithm presented in Section 5.5. The system, consisting of $N = 5$ was initialised as a strongly connected graph according to Fig. 14 and system matrix $(A_I - M)$ according to (6.7) and Table 6. Additionally, the constraints where set up as (6.8).



(a) Estimated and true values of the vector u for the original system.

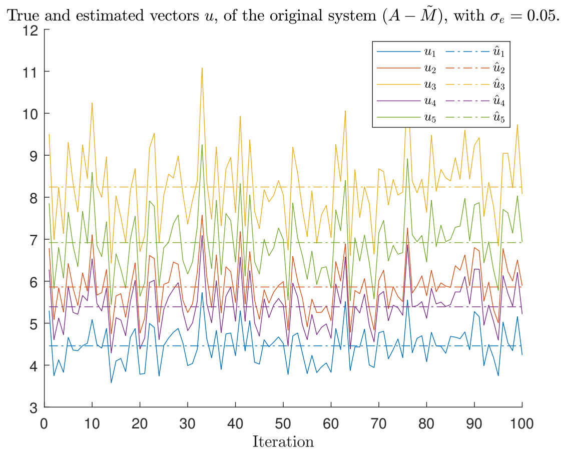


(b) Estimated and true values of the vector u for the adapted system.

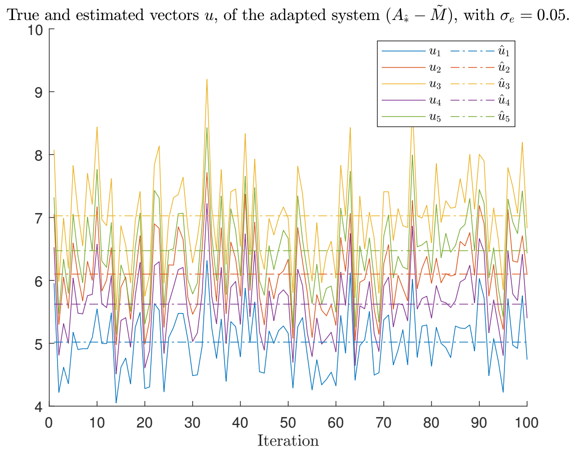


(c) Estimated and true values of the vector u for the optimal system.

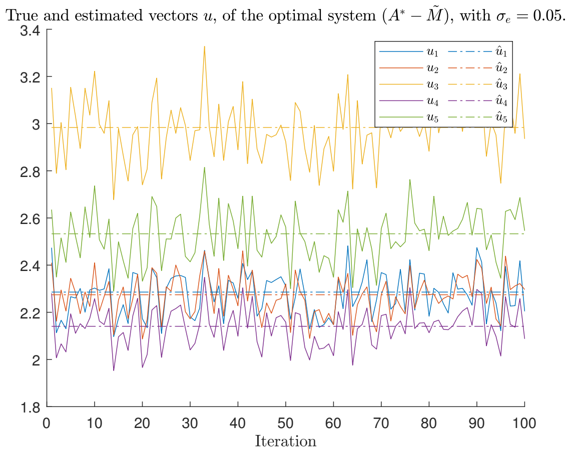
Figure 12: Estimated and true values of the vector u for respective system during simulations of control model errors.



(a) Estimated and true values of the vector u for the original system.



(b) Estimated and true values of the vector u for the adapted system.



(c) Estimated and true values of the vector u for the optimal system.

Figure 13: Estimated and true values of the vector u for respective system during simulations of connection model errors.

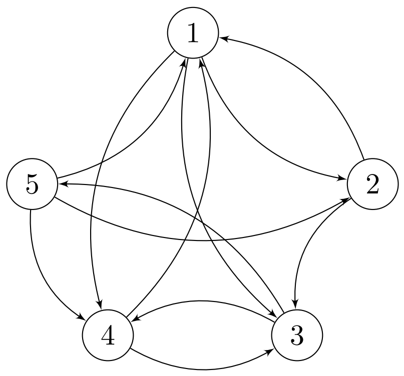


Figure 14: Graph of the strongly connected initialised network.

Table 6: Control parameters for the original, locally adapted and optimal systems

A_I	$\hat{A}_*$	A_*
1.9493	2.3174	2.6996
1.9493	1.9290	2.6996
1.9493	2.4771	2.6996
1.9493	1.8740	2.6996
1.9493	1.8750	1.8751

Table 7: Measured total distance between disturbed and undisturbed trajectories, together with respective systems u.

Original system		Adapted system		Optimal system	
u	Δx	$\hat{u}^*$	Δx	u^*	Δx
13.8048	30.1310	4.7620	12.7054	1.6688	4.7827
11.8970	26.3698	4.6691	12.7888	1.6110	4.7672
17.5946	37.1469	5.3312	13.3432	1.9523	4.8063
10.8117	22.7121	4.1365	10.1335	1.4528	3.4724
4.4883	9.0539	1.7856	3.9255	0.9919	1.95288

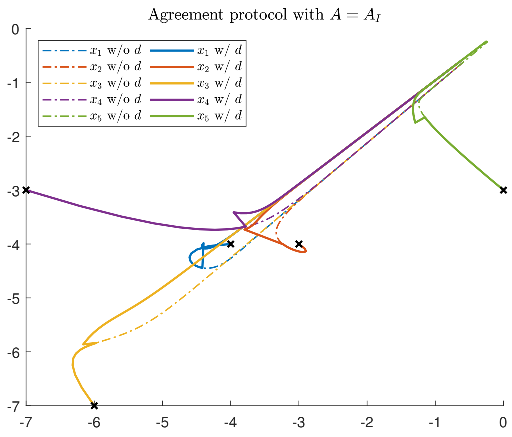
$$M = \begin{bmatrix} 0 & -1.0298 & 0 & -1.2519 & -0.0274 \\ -1.1252 & 0 & 0 & 0 & -1.4833 \\ -1.1222 & -0.4207 & 0 & -1.1839 & 0 \\ -0.1584 & 0 & -0.6718 & 0 & -1.3629 \\ 0 & 0 & -0.4404 & 0 & 0 \end{bmatrix} \tag{6.7}$$

$$\underline{a}_i = 1.01\rho(M),\ \bar{a}_i = 0.99 \max_i \left(\sum_{j \in \mathcal{N}_i^{\text{in}}} m_{ij} \right) \tag{6.8}$$

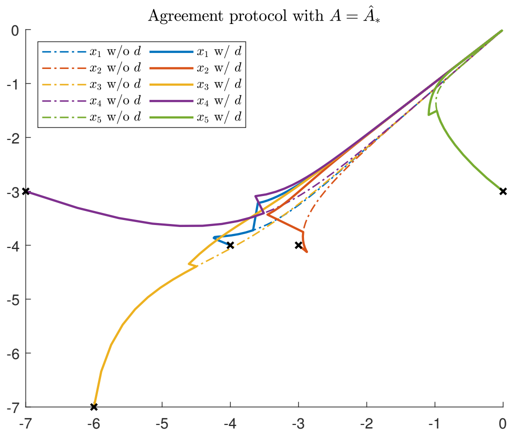
As the algorithm (5.19), had converged for the initial system, the adapted system was set up with control parameters according to $(\hat{A}_* - M)$, from Table 6 and (6.7). To compare this to a potentially better system, a system was set up as $(A_* - M)$, from Table 6 and (6.7), for which the control parameters were found through the use of Matlab built-in solver 'fmincon' on the centralised problem, and therefore seen as the optimal system. Further, the disturbance was modelled such that at time $t = 1$s there was a random disturbance acting on the system for the duration of $\Delta t = 0.1$s, the same disturbance was used for both simulations. The trajectories of both system, including the trajectories of the systems without disturbance, can be seen in Fig. 15.

As a measurement of how much the systems where disturbed the distance, Δx, between the undisturbed and the disturbed trajectory was measured at each $\Delta t = 0.1$s and the total distance was calculated as the sum of these, this can be seen together with respective systems vector u in Table 7.

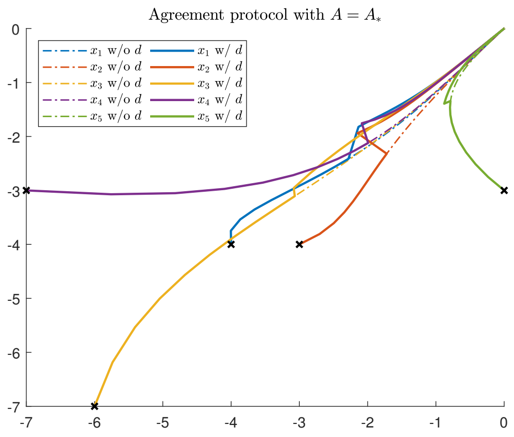
Even though the distributed algorithm achieves both reducing the overall bound γ and most of the agents individual control effort, we see that it does increase the control effort for two of the agents. We can see that this is also the case for the optimal system. However, in the system constructed by $\tilde{A} = A_*$, only one agent sees a reduction in control effort but the overall bound γ is reduced even farther. Although, choosing other weights α and β could shift this, such that the reduction of overall bound γ was reduced. Further, the algorithm does not take into account



(a) Trajectories of the original system, the undisturbed trajectories can be seen as dashed lines for comparison.



(b) Trajectories of the adapted system, the undisturbed trajectories can be seen as dashed lines for comparison.



(c) Trajectories of the optimal system, the undisturbed trajectories can be seen as dashed lines for comparison.

Figure 15: Undisturbed and disturbed trajectories of the three simulated systems.

control costs or any future changes to the edges, such that some agents could see their control effort increased to their upper bound, $\bar{a}_i$, which could in turn be problematic. If the weight on the incoming edges for those agents were to increase or if new edges where connected, as there would be no headroom, such an increase in edge weights would lead to a decrease in network performance.

Such an aspect could be treated by adding a term which penalizes large control effort to the local objective function in (5.18). As of Lemma 5, u_i as a function of control parameters is a convex function, which should indicate that (5.18) could be made into a convex function of a_i. However, this avenue has not been explored, since it is uncertain if the approach of each agent minimizing its objective function greedily, can ensure global convergence.

6.4 Generalisation of fairness

Throughout the thesis we have limited our results and discussion to linear scalar systems, with some extensions available as an outlook in the appendix. However, in a more general system it may be so that the agents, and the constraints imposed on an agent, are more heterogeneous. Such that one agent could have constraints limiting how it can choose its control parameters, while other agents have restrictions on their weights or connections, or constraints in properties we have not even considered. In these situations, as well as for future research, one could consider the situation where the agents can implement some cost function on the properties and that the notion of χ-fairness and measurement of unfairness are instead used to find a fair cost. This scenario could also allow for the situation where an agent cannot increase network performance or decrease cost, in one property, but can instead choose to do it in the parameters for which this is a possibility. Before researching this avenue, the results on fairness will most likely need to be extended for more properties and non-scalar agents, but with more research it could be possible to extend the notion of fairness from its somewhat one-dimensional applicability it currently has, to give a more detailed image of the systems.

7 Conclusion

In this master thesis we have investigated how the distribution of control effort in large-scale interconnected systems affects network performance. This has resulted in some new knowledge on how the distribution of bounds on state deviations, within a network, behaves as a function of control parameters. In order to capture how this distribution relates to control effort we have introduced the notions of *fairness* and measurement of *unfairness*. Further, given some assumptions on constraints that ensures stability and connectivity of networks, we have presented decentralised algorithms. While these might not achieve a globally optimal solution to the problem, the algorithms do ensure an improved network performance, given the chosen measurements. In addition to this we have presented some simulations of how the results hold, when measurement and model errors are present. These simulations do not give a full picture, but can hopefully indicate some robustness of the results.

As the results heavily rely on the assumption that each agent knows a lower bound on control effort that still ensures stability, more research into finding this bound, in a decentralised fashion, would be useful.

Future work can include decentralised algorithms that can find the optimal solution to the problem at hand, as well as decentralised algorithms to find bounds of control effort that ensures stability. Future work could also include extending the results for a broader range of agents and properties, along with the discussion points brought up earlier.

Appendix

A. Non-scalar systems

Some results that has come up during this project relates to non-scalar systems. However, these results are somewhat scattered and do not relate to general non-scalar systems or general strategies. Therefore we have included these results in the appendix as an outlook on future work within γ-robustness and χ-fairness.

System and network structure

We start by defining how the network is structured when the agents are represented as general systems, rather than scalar systems. All agents are therefore defined as follows

$$\dot{x}_i = (-\alpha_i A_i)x_i, \quad x_i \in \mathbb{R}^{n_i},\, A_i \in \mathbb{R}^{n_i \times n_i}$$

with α_i as a diagonal matrix $\alpha_i \in \mathbb{R}^{n_i \times n_i}$, $\alpha_i > 0$. This system is then connected to its neighbours in the set $\mathcal{N}_i$ as

$$\Sigma_i : \quad \dot{x}_i = -\alpha_i A_i x_i + \sum_{j \in \mathcal{N}_i} M_{ij} x_j + d_i$$

where $M_{ij} \in \mathbb{R}^{n_i \times n_j}$, $M_{ij} \geq 0$.
The entire system can then be described as

$$\Sigma : \quad \dot{x} = -(\alpha A - M)x + d \tag{A.1}$$

where

$$\alpha = \begin{bmatrix} \alpha_1 & & 0 \\ & \ddots & \\ 0 & & \alpha_n \end{bmatrix} A = \begin{bmatrix} A_1 & & 0 \\ & \ddots & \\ 0 & & A_N \end{bmatrix},\ M = \begin{bmatrix} 0 & M_{12} & \dots \\ M_{21} & \ddots & \vdots \\ \vdots & \dots & 0 \end{bmatrix}.$$

Representing the network in this structure allows to choose control parameters such that every state could be treated as its own agent.

Conditions for γ-robustness

In the following sections we will only consider agents which are asymptotically stable, since we are interested in the scalability and robustness of interconnected systems.

We investigate what conditions and what strategies may exist when the self-feedback parameters can be chosen freely. A system is γ-robust if and only if there exists a $v \in \mathbb{R}^N$, $v > 0$ such that (4.4) is fulfilled. Since this has to hold for all rows, we can start by investigating under which conditions it holds for Σ_i. We start by assuming that the system is asymptotically stable and that there exists such a v and get the following

$$\Sigma_i : -\alpha_i A_i v_i + \sum_{j \in \mathcal{N}_i} M_{ij} v_j + \mathbb{1} \leq 0 \tag{A.2}$$

where v_i and v_j are vectors containing the elements of v that corresponds to agents i and j respectively. During the project the following classes of matrices have been found to be plausible to form sufficient conditions on A_i to guarantee the existence of a suitable α_i, such that the overall system is γ-robust.

1. A_i is a non-singular M-matrix.
2. A_i is inverse-positive, meaning $A_i^{-1} \geq 0$.
3. A_i is non-negative, with at least one positive element in each row.

Adding a node with matrix A_{N+1} being M-matrix or inverse-positive.

Proposition 16. *Let Σ be a γ-robust system that fulfils* (4.4)*, adding the node $N+1$, when the matrix A_{N+1} is a M-matrix, can be made γ-scalable by choosing an α_{N+1} such that $\alpha_{N+1}A_{N+1}\mathbb{1} \geq \frac{1}{\gamma}\mathbb{1}$.*

Proof. For the new system to be γ-robust, there must exist a vector $u_{N+1} > 0$ such that

$$\bar{\Sigma} : -\begin{bmatrix} A - M & \\ & \alpha_{N+1}A_{N+1} \end{bmatrix} \begin{bmatrix} u \\ u_{N+1} \end{bmatrix} = -\mathbb{1}. \tag{A.3}$$

If A_{N+1} is a M-matrix, then it will also be inverse positive and there will exist a u_{N+1} such that $0 < u_{N+1} \leq \gamma\mathbb{1}$ as

$$u_{N+1} = (\alpha_{N+1}A_{N+1})^{-1}\mathbb{1} = A_{N+1}^{-1} \begin{bmatrix} \frac{1}{\alpha_{N+1_1}} \\ \vdots \\ \frac{1}{\alpha_{N+1_n}} \end{bmatrix}. \tag{A.4}$$

Since both $A_{N+1}^{-1} \geq 0$ and $\alpha_{N+1} \geq 0$ it will be possible to choose an α_{N+1} such that $0 < u_{N+1} \leq \gamma\mathbb{1}$. □

In fact, this proof may hold for any inverse-positive matrix. However, depending on constraints on control parameters, and if an agent can choose control parameters freely for all its states, choosing control parameters may need to be solved by some optimisation method.

Adding an edge to a system with matrix A_i being inverse-positive

Without loss of generality we choose adding the edge (i_1, l) from agent l to the first state of system i, now assuming that A_i is inverse-positive, then adding an edge can be made γ-scalable if the following is satisfied

$$-\alpha_i A_i u_i + \begin{bmatrix} \sum_{j\in\mathcal{N}_i}(M_{ij})_{1\bullet}u_j + m_{i_1 l}u_l \\ \sum_{j\in\mathcal{N}_i}(M_{ij})_{2\bullet}u_j \end{bmatrix} + \mathbb{1} \leq 0 \text{ , } u_i \leq \gamma\mathbb{1}. \tag{A.5}$$

That this is possible to fulfil can be shown by

$$\begin{aligned} -\alpha_i A_i u_i + \begin{bmatrix} \sum_{j\in\mathcal{N}_i}(M_{ij})_{1\bullet}u_j + (M_{il})_{1\bullet}u_l \\ \sum_{j\in\mathcal{N}_i}(M_{ij})_{2\bullet}u_j \end{bmatrix} + \mathbb{1} \leq 0 \Rightarrow \\ \alpha_i A_i u_i \geq \mathbb{1} + \begin{bmatrix} \sum_{j\in\mathcal{N}_i}(M_{ij})_{1\bullet}u_j + (M_{il})_{1\bullet}u_l \\ \sum_{j\in\mathcal{N}_i}(M_{ij})_{2\bullet}u_j \end{bmatrix} \Rightarrow \\ u_i \geq A_i^{-1} \begin{bmatrix} \frac{1+\sum_{j\in\mathcal{N}_i}(M_{ij})_{1\bullet}u_j + (M_{il})_{1\bullet}u_l}{\alpha_{i_1}} \\ \frac{1+\sum_{j\in\mathcal{N}_i}(M_{ij})_{2\bullet}u_j}{\alpha_{i_n}} \end{bmatrix}. \end{aligned} \tag{A.6}$$

The equation above also shows that, for systems with inverse-positive matrices, it is necessary to only change the element of α_i which corresponds to the added edge. But, in the same manner as when adding a node, it may not be possible or desired to only alter one of the elements in α_i. We can also see that finding this vector u may require some different computations compared to the case with only scalar agents. Although this shows that it is possible to add an edge to a system with A_i being inverse-positive, there may be additional conditions to ensure stability when there are both incoming and outgoing edges from such systems.

Adding a node with matrix A_{N+1} being non-negative

Proposition 17. *Let A_{N+1} be a non-negative matrix with at least one positive element in each row. Adding a node with this A_{N+1} can be made scalable if a self-feedback α_{N+1} is chosen such that*

$$\bar{\Sigma} : -\begin{bmatrix} A - M & \\ & \alpha_{N+1}A_{N+1} \end{bmatrix} \begin{bmatrix} v \\ v_{N+1} \end{bmatrix} \leq -1,\ 0 < \bar{v} \leq \gamma\mathbb{1}.$$

Proof. Since A_{N+1} is non-negative $A_{N+1}v_{N+1} > 0$ for all $v_{N+1} > 0$, adding this node will be a γ-scalable change by choosing an $\alpha_{N+1} > 0$ such that $-A_{N+1}v_{N+1} \leq -\alpha_{N+1}^{-1}\mathbb{1}$ which can be satisfied for a vector $v_{N+1} \leq \gamma\mathbb{1}$ by choosing an $\alpha_{N+1}^{-1}\mathbb{1} \leq \gamma A_{N+1}\mathbb{1}$. $\square$

Choosing α_{N+1} can be done by considering the constraints in control parameters in the same manner as when A_{N+1} is M-matrix or inverse-positive.

Adding an edge to a system with matrix A_i being non-negative

Without loss of generality we choose adding the edge (i_1, l) from agent l to the first state of system i, now assuming that A_i is non-negative with at least one positive element in each row, then adding an edge can be made γ-scalable if the following is satisfied

$$-\alpha_i A_i u_i + \begin{bmatrix} \sum_{j\in\mathcal{N}_i}(M_{ij})_{1\bullet}u_j + (M_{il})_{1\bullet}u_l \\ \sum_{j\in\mathcal{N}_i}(M_{ij})_{2\bullet}u_j \end{bmatrix} + \mathbb{1} \leq 0\ , u_i \leq \gamma\mathbb{1}. \tag{A.7}$$

That this is possible to fulfil can be shown by

$$\begin{aligned} -\alpha_i A_i u_i + \begin{bmatrix} \sum_{j\in\mathcal{N}_i}(M_{ij})_{1\bullet}u_j + (M_{il})_{1\bullet}u_l \\ \sum_{j\in\mathcal{N}_i}(M_{ij})_{2\bullet}u_j \end{bmatrix} + \mathbb{1} \leq 0 \Rightarrow \\ \alpha_i A_i u_i \geq \mathbb{1} + \begin{bmatrix} \sum_{j\in\mathcal{N}_i}(M_{ij})_{1\bullet}u_j + (M_{il})_{1\bullet}u_l \\ \sum_{j\in\mathcal{N}_i}(M_{ij})_{2\bullet}u_j \end{bmatrix} \Rightarrow \\ A_i u_i \geq \begin{bmatrix} \frac{1+\sum_{j\in\mathcal{N}_i}(M_{ij})_{1\bullet}u_j+(M_{il})_{1\bullet}u_l}{\alpha_{i_1}} \\ \frac{1+\sum_{j\in\mathcal{N}_i}(M_{ij})_{2\bullet}u_j}{\alpha_{i_n}} \end{bmatrix}. \end{aligned} \tag{A.8}$$

Since A_i is non-negative and u_i is positive, $A_i u_i > 0$ and it will be possible to choose an α_i such that the inequality holds. However, finding this vector u may require some different computations compared to the case with only scalar agents. Although this shows that it is possible to add an edge to a system with A_i being non-negative, there may be additional conditions to ensure stability when there are both incoming and outgoing edges from such systems.

References

[1] G. James et al. "A Deployed Multi-agent Framework for Distributed Energy Applications". In: *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems* (2006).

[2] B. Cheng and H.H. Cheng. “A Review of the Applications of Agent Technology in Traffic and Transportation Systems”. In: *IEEE Transactions on Intelligent Transportation Systems* (2010).

[3] Daniel Pickem et al. “The robotarium: A remotely accessible swarm robotics research testbed”. In: *IEEE International Conference on Robotics and Automation* (2017).

[4] V. C. Gungor, B. Lu, and G. H. Hancke. “Opportunities and Challenges of Wireless Sensor Networks in Smart Grid”. In: *IEEE Transactions on Industrial Electronics* (2010).

[5] S. Knorn et al. “Distortion Minimization in Multi-Sensor Estimation Using Energy Harvesting and Energy Sharing”. In: *IEEE Transactions on Signal Processing* (2015).

[6] A. Alam et al. “Heavy-duty vehicle platooning towards sustainable freight transportation: A cooperative method to enhance safety and efficiency”. In: *IEEE Control Systems Magazine* 35.6 (2015), pp. 34–56.

[7] D.D. Šiljak. *Large-scale dynamic systems: stability and structure.* North-Holland Books, New York, USA, 1978.

[8] P.J. Moylan and D.J. Hill. “Stability criteria for large-scale systems”. In: *IEEE Transactions on Automatic Control* AC-23.2 (1978), pp. 143–149.

[9] D. Swaroop and J.K. Hedrick. “String stability of interconnected systems”. In: *IEEE Transactions on Automatic Control* (1996).

[10] S. Knorn and B. Besselink. “Scalable robustness of interconnected systems subject to structural changes”. In: *IFAC World Congress.* 2020.

[11] A. Chapman, E. Schoof, and M. Mesbahi. “Distributed Online Topology Design for Network-level Disturbance Rejection”. In: *52nd IEEE Conference in Decision and Control* (2013).

[12] R.A. Horn and C.R. Johnson. *Matrix analysis.* 2nd. Cambridge University Press, Cambridge, UK, 2013.

[13] R.A. Horn and C.R. Johnson. *Topics in matrix analysis.* Cambridge Univsersity Press, 1991.

[14] A. Berman and R.J. Plemmons. *Nonnegative matrices in the mathematical sciences.* SIAM, Philadelphia, USA, 1994.

[15] C.R. Johnson and R.L. Smith. “Inverse M-matrices, II”. In: *Linear Algebra and Its Applocations* (2011).

[16] R. Diestel. *Graph Theory.* Springer-Verlag, Heidelberg, 2016.

[17] M. Mesbahi and M. Egerstedt. *Graph theoretic methods in multiagent Networks.* Princeton University Press, 2010.

[18] R.T. Rockefellar. *Convex Analysis.* Princeton University Press, 1970.

[19] I. Griva, S.G. Nash, and A. Sofer. *Linear and nonlinear optimization.* SIAM, 2009.

[20] D.P. Bertsekas. “On the Goldstein-Levitin-Polyak Gradient Projection Method”. In: *IEEE Trans. Automatic Control* (1976).

[21] D. Liberzon and A.S. Morse. “Basic Problems in Stability and Design of Switched Systems”. In: *IEEE Control Systems Magazine* (1999).

[22] S. Liu J. Mou and S. Morse. “Asycnhronous Distributed Algorithms for Solving Linear Algebraic Equations”. In: *IEEE Trans. Autom. Control* (2018).

[23] M. Axelson-Fisk and S. Knorn. “Aspects of Fairness in Robust, Distributed Control of Interconnected Systems”. Submitted to: 59th Conference on Decision and Control. 2020.

[24] A. Nedić and A. Ozdaglar. “Distributed Subgradient Methods for Multi-Agent Optimization”. In: *IEEE Transactions on Automatic Control* (2009).

[25] A. Nedić, A. Ozdaglar, and P.A. Parrilo. “Constrained Consensus and Optimization in Multi-Agent Networks”. In: *IEEE Trans. Autom. Control* (2010).

[26] D. Yuan, S. Xu, and H. Zhao. “Distributed Primal–Dual Subgradient Method forMultiagent Optimization via Consensus Algorithms”. In: *IEEE Trans. Systems, Man, and Cybernetics, Vol 41* (2011).

[27] A. Chapman and M. Mesbahi. “Stability Analysis of Nonlinear Networks via M-matrix Theory: Beyond Linear Consensus”. In: *American Control Conference* (2012).