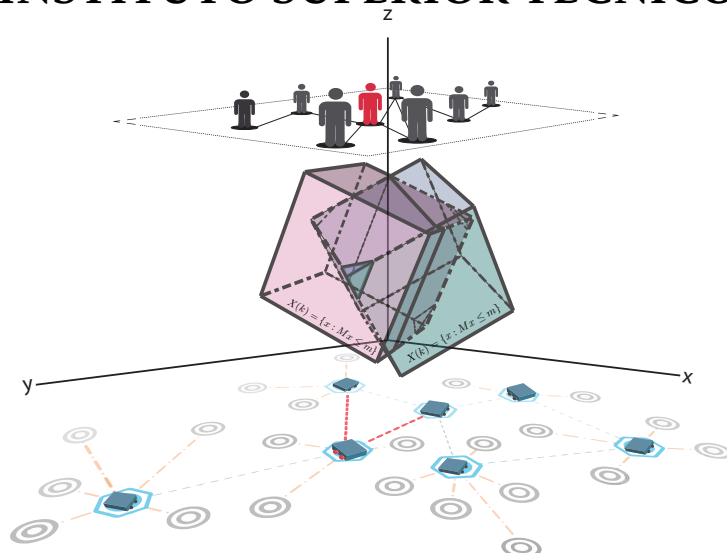


**UNIVERSIDADE DE LISBOA**  
**INSTITUTO SUPERIOR TÉCNICO**



**Fault-tolerant Stochastic Distributed Systems**

**Daniel de Matos Silvestre**

**Supervisor:** Doctor Carlos Jorge Ferreira Silvestre

**Co-Supervisor:** Doctor João Pedro Cordeiro Pereira Botelho Hespanha

**Thesis approved in public session to obtain the PhD Degree in  
Electrical and Computer Engineering**

**Jury final classification: Pass with Distinction and Honour**



**UNIVERSIDADE DE LISBOA**  
**INSTITUTO SUPERIOR TÉCNICO**

**Fault-tolerant Stochastic Distributed Systems**

**Daniel de Matos Silvestre**

**Supervisor:** Doctor Carlos Jorge Ferreira Silvestre  
**Co-Supervisor:** Doctor João Pedro Cordeiro Pereira Botelho Hespanha

**Thesis approved in public session to obtain the PhD Degree in  
Electrical and Computer Engineering**

**Jury final classification: Pass with Distinction and Honour**

**Jury**

**Chairperson:** Doctor José Alberto Rosado dos Santos Victor  
Instituto Superior Técnico, Universidade de Lisboa

**Members of the committee:**

Doctor Sandro Zampieri  
Università degli Studi di Padova, Italy  
Doctor Luís Miguel Teixeira D'Ávila Pinto da Silveira  
Instituto Superior Técnico, Universidade de Lisboa  
Doctor António Pedro Rodrigues Aguiar  
Faculdade de Engenharia da Universidade do Porto  
Doctor Carlos Jorge Ferreira Silvestre  
Instituto Superior Técnico, Universidade de Lisboa  
Doctor Paulo Jorge Coelho Ramalho Oliveira  
Instituto Superior Técnico, Universidade de Lisboa  
Doctor João Manuel de Freitas Xavier  
Instituto Superior Técnico, Universidade de Lisboa

**Funding Institution:**

Fundação para a Ciência e a Tecnologia



“The Science of today is the technology of tomorrow.”  
– Edward Teller, *The Legacy of Hiroshima* (1962), 146.



The present doctoral thesis discusses the design of fault-tolerant distributed systems, placing emphasis in addressing the case where the actions of the nodes or their interactions are stochastic. The main objective is to detect and identify faults to improve the resilience of distributed systems to *crash-type* faults, as well as detecting the presence of malicious nodes in pursuit of exploiting the network. The proposed analysis considers malicious agents and computational solutions to detect faults.

Crash-type faults, where the affected component ceases to perform its task, are tackled in this thesis by introducing stochastic decisions in deterministic distributed algorithms. Prime importance is placed on providing guarantees and rates of convergence for the steady-state solution. The scenarios of a social network (state-dependent example) and consensus (time-dependent example) are addressed, proving convergence. The proposed algorithms are capable of dealing with packet drops, delays, medium access competition, and, in particular, nodes failing and/or losing network connectivity.

The concept of Set-Valued Observers (SVOs) is used as a tool to detect faults in a worst-case scenario, i.e., when a malicious agent can select the most unfavorable sequence of communications and inject a signal of arbitrary magnitude. For other types of faults, it is introduced the concept of Stochastic Set-Valued Observers (SSVOs) which produce a confidence set where the state is known to belong with at least a pre-specified probability. It is shown how, for an algorithm of consensus, it is possible to exploit the structure of the problem to reduce the computational complexity of the solution. The main result allows discarding interactions in the model that do not contribute to the produced estimates.

The main drawback of using classical SVOs for fault detection is their computational burden. By resorting to a left-coprime factorization for Linear Parameter-Varying (LPV) systems, it is shown how to reduce the computational complexity. By appropriately selecting the factorization, it is possible to consider detectable systems (i.e., unobservable systems where the unobservable component is stable). Such a result plays a key role in the domain of Cyber-Physical Systems (CPSs). These techniques are complemented with Event- and Self-triggered sampling strategies that enable fewer sensor updates. Moreover, the same triggering mechanisms can be used to make decisions of when to run the SVO routine or resort to over-approximations that temporarily compromise accuracy to gain in performance but maintaining the convergence characteristics of the set-valued estimates. A less stringent requirement for network resources that is vital to guarantee the applicability of SVO-based fault detection in the domain of Networked Control

## Abstract

---

Systems (NCSs).

**Keywords:** Fault-tolerant; Distributed Algorithms; Networked Control Systems; Set-valued Observers; Event- and Self-triggered Systems.



A presente tese de doutoramento desenvolve técnicas de projecto de sistemas distribuídos tolerantes a falhas, focando em particular algoritmos nos quais as acções de cada nó e as interacções entre nós têm carácter estocástico. O objectivo principal é detectar e identificar falhas por forma a melhorar a tolerância a falhas do tipo crash em sistemas distribuídos, bem como detectar a presença de agentes maliciosos à procura de explorar e tomar o controlo do sistema. A análise proposta considera agentes maliciosos e soluções computacionais utilizáveis no contexto da detecção de falhas.

No presente estudo, abordam-se falhas do tipo crash, onde o componente afectado pela falha deixa de funcionar completamente, que são tratadas através da introdução de decisões estocásticas em sistemas determinísticos distribuídos. O objectivo da análise é garantir a convergência bem como determinar a velocidade a que o sistema atinge a solução estacionária. O caso de uma rede social (exemplo de dinâmica dependente do estado) e de um algoritmo de consenso (dinâmica dependente do tempo) são estudados, sendo provada convergência, tornando-os robustos à perda de pacotes na rede, atrasos, competição por acesso ao meio partilhado e, em particular, a agentes que deixam de funcionar e/ou perdem conectividade.

Para um modelo de falhas mais genérico que o tipo crash, este trabalho recorre a Set-Valued Observers (SVOs) como ferramenta para detectar falhas no pior cenário, i.e., quando um agente malicioso pode seleccionar a sequência de comunicações mais desfavorável e injectar um sinal de magnitude arbitrária. Para outros tipos de falhas, em que os nós não se comportam de acordo com as distribuições de probabilidade do modelo, é introduzido o conceito de Stochastic Set-Valued Observers (SSVOs) que produzem um intervalo de confiança que contém o estado do sistema com uma probabilidade pré-definida. Para um algoritmo de consenso é demonstrado como é possível explorar a estrutura do problema de forma a diminuir a complexidade computacional da solução. O resultado principal é a remoção no modelo das interacções que não têm impacto nos conjuntos estimados.

A principal desvantagem dos SVOs clássicos no contexto de detecção de falhas é o seu peso computacional. Recorrendo a uma factorização coprima à esquerda, para sistemas lineares com parâmetros variantes no tempo, mostra-se como reduzir a sua complexidade computacional. Selecionando apropriadamente a factorização é possível ainda considerar sistemas detectáveis (i.e., sistemas não observáveis mas cuja componente não observável é estável). Este resultado é de particular importância no domínio dos Cyber-Physical Systems (CPSs). Estas técnicas são complementadas com estratégias do tipo event- e self-triggered que permitem reduzir a frequência de envio das medidas dos sensores. As mesmas podem ser utilizadas para tomar decisões de quando executar a rotina dos SVOs ou utilizar aproximações, comprometendo a precisão, para ganhar em tempo computacional, mantendo a convergência das estimativas

## Resumo

---

destes observadores. O desenvolvimento desta estratégias é fundamental uma vez que a redução de utilização dos recursos da rede é essencial para garantir a aplicabilidade da detecção de falhas com base em SVOs no domínio dos Networked Control Systems (NCSs).

**Palavras-chave:** Tolerância a Falhas; Algoritmos Distribuídos; Sistemas de Controlo em Rede; Observadores com Conjuntos; Sistemas auto-despoletados ou por eventos.

To my family.



## ACKNOWLEDGMENTS

My first words of utmost appreciation go to my advisors, Professor Carlos Silvestre and Professor João Hespanha, for their help, support and guidance from the beginning of my research work. I am grateful for their contributions in driving me to excel and improve my research methodology while encouraging me to be creative. It is my deep belief that all the discussions and debates of the last five years have led me to be more assertive and to develop a way of thinking based on solid scientific foundations. Their comments about my work have motivated me to always question my own conclusions and see the small challenges that sometimes hinder in the details. I have no doubt they have made a serious impact on my academic path and also in my personal life.

I would also like to express my gratitude to the remaining members of the committee that evaluated my progress, namely Professor João Xavier for his comments and insights about many topics that led me to a better understanding of the mathematical machinery behind the results. I could not forget Professor Paulo Oliveira for the many discussions about a whole myriad of topics.

I am extremely thankful to Paulo Rosa for the friendship and all the fruitful and pleasant discussion on all sorts of topics. I will forever remember and appreciate your help along most of the years in my PhD. I could not forget Duarte Antunes and Rita Cunha for their contributions and guidance in the early stages my research.

My thanks go also to all my friends and colleagues, especially those at the Electrical and Computer Engineering department at IST and at ISR. In particular, I would like to thank Sérgio Brás, Tiago Gaspar, João Almeida, Pedro Casau, Daniel Viegas and Pedro Lourenço for their support and friendship and for turning these years so memorable. I will treasure and cherish all the good times.

At last, but not least, I am grateful to my family for unconditionally support, endless love and tolerance. I will be forever in your debt for being such a foundation in my life.

**Financial support:** This work was partially funded by the project FCT [UID/EEA/50009/2013] and with grant SFRH//BD/71206/2010, from Fundação para a Ciência e a Tecnologia.



<b>Abstract</b>	<b>vii</b>
<b>Resumo</b>	<b>ix</b>
<b>Acknowledgments</b>	<b>xiii</b>
<b>Contents</b>	<b>xv</b>
<b>List of Figures</b>	<b>xix</b>
<b>List of Tables</b>	<b>xxiii</b>
<b>List of Theorems</b>	<b>xxv</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Previous Work and Brief Literature Review . . . . .	3
1.3 Contributions of the Thesis . . . . .	5
1.4 Organization of the Thesis . . . . .	6
1.4.1 Randomized Time-Dependent Algorithms . . . . .	6
1.4.2 Randomized State-Dependent Algorithms . . . . .	6
1.4.3 Set-Valued Estimators . . . . .	7
1.4.4 SVOs for LPV systems with Coprime Factorization . . . . .	7
1.4.5 Fault Detection and Isolation in Detectable Systems . . . . .	8
1.4.6 Event- and Self-Triggered NCS and Set-Valued Observers . . . . .	8
1.5 Notations and Definitions . . . . .	9
<b>2 Randomized Time-Dependent Algorithms</b>	<b>11</b>
2.1 Introduction . . . . .	11
2.2 Main Contributions and Organization . . . . .	11
2.3 Consensus . . . . .	12
2.3.1 Problem Description . . . . .	14
2.3.2 Proposed Solution . . . . .	15
2.4 Convergence Analysis . . . . .	17
2.5 Convergence Rates . . . . .	27
2.5.1 Distributed Optimization . . . . .	28
2.5.2 Comparison between unidirectional and bidirectional case . . . . .	29
2.6 Conclusions . . . . .	30

## Contents

---

<b>3</b>	<b>Randomized State-Dependent Algorithms</b>	<b>33</b>
3.1	Introduction . . . . .	33
3.2	Main Contributions and Organization . . . . .	33
3.3	Social Networks . . . . .	34
3.3.1	Motivation . . . . .	34
3.3.2	Related Work . . . . .	35
3.3.3	Problem Statement . . . . .	38
3.4	Neighbor Selection Rules . . . . .	39
3.5	Stochastic State-Dependent Social Network . . . . .	43
3.6	Main Properties . . . . .	44
3.6.1	Deterministic Social Network . . . . .	44
3.6.2	Base Network . . . . .	46
3.6.3	Nearest Distinct Values . . . . .	50
3.6.4	Nearest Circular Value . . . . .	52
3.6.5	Nearest Distinct Neighbors . . . . .	53
3.6.6	Stochastic Social Network . . . . .	56
3.7	Simulation Results . . . . .	63
3.8	Conclusions . . . . .	68
<b>4</b>	<b>Set-Valued Estimators</b>	<b>71</b>
4.1	Introduction . . . . .	71
4.2	Main Contributions and Organization . . . . .	74
4.3	Fault Detection Problem . . . . .	75
4.4	Fault Detection using Set-Valued Observers (SVOs) . . . . .	77
4.5	Fault Detection using Stochastic Set-Valued Observers (SSVO) . . . . .	86
4.6	Byzantine Consensus Algorithm . . . . .	89
4.7	Theoretical overbound on the fault signal . . . . .	94
4.8	Asymptotic correctness . . . . .	97
4.9	Application of Set Estimators to Set Consensus . . . . .	101
4.9.1	Broadcast solution using position . . . . .	103
4.9.2	Unicast solution using estimation . . . . .	104
4.9.3	Convergence to Set-consensus . . . . .	105
4.10	Simulation Results . . . . .	106
4.11	Conclusions . . . . .	116
<b>5</b>	<b>Coprime Factorization</b>	<b>119</b>
5.1	Introduction . . . . .	119
5.2	Main Contributions and Organization . . . . .	120
5.3	Problem Statement . . . . .	121



5.4	Deadbeat Observers for LPV systems . . . . .	124
5.5	Coprime Factorization . . . . .	125
5.6	Fault Detection . . . . .	126
5.7	Fault Isolation . . . . .	129
5.8	Example and Simulations . . . . .	131
5.9	Conclusions . . . . .	136
<b>6</b>	<b>FDI in Detectable Systems</b>	<b>137</b>
6.1	Introduction . . . . .	137
6.2	Main Contributions and Organization . . . . .	139
6.3	Observability issue . . . . .	140
6.3.1	Systems of Systems . . . . .	140
6.3.2	Smart Grids . . . . .	141
6.4	SVOs for detectable systems . . . . .	144
6.5	Fast SVOs . . . . .	146
6.6	Simulation Results . . . . .	149
6.7	Conclusions . . . . .	158
<b>7</b>	<b>Event- and Self-Triggered strategies</b>	<b>159</b>
7.1	Introduction . . . . .	159
7.2	Main Contributions and Organization . . . . .	162
7.3	Problem Statement . . . . .	163
7.4	Set-valued Estimate Approximations . . . . .	165
7.4.1	Hyper-parallelepiped Approximation . . . . .	165
7.4.2	Ellipsoidal Overbounding . . . . .	170
7.5	Set-Valued Observer for Event- and Self-Triggered Systems . . . . .	171
7.5.1	Set-Valued Observers for Event-Triggered Systems . . . . .	172
7.5.2	Set-Valued Observers for Self-Triggered Systems . . . . .	174
7.6	Event- and Self-Triggered Set-Valued Observers . . . . .	175
7.6.1	Event-Triggered Set-Valued Observers . . . . .	177
7.6.2	Self-Triggered Set-Valued Observers . . . . .	179
7.6.3	Distributed Systems . . . . .	180
7.7	Triggering Frequency and Convergence . . . . .	183
7.7.1	Worst-case Scenario . . . . .	184
7.7.2	Stochastic case . . . . .	185
7.8	Simulation Results . . . . .	188
7.9	Conclusions . . . . .	193

---

**Contents**

---

<b>8</b>	<b>Conclusions and Future Directions</b>	<b>195</b>
8.1	Future Directions . . . . .	197
<b>A</b>	<b>Appendix</b>	<b>199</b>
	<b>Bibliography</b>	<b>201</b>

# LIST OF FIGURES

2.1	Communication graph with different out-neighbor degrees. . . . .	29
3.1	Network generated for each definition using $\eta = 1$ and $x_1 = 1, x_2 = 2, x_3 = 3, x_4 = 3$ and $x_5 = 4$ . . . . .	42
3.2	Detail of the links from node $x_3$ when using $\eta = 2$ and $x_1 = 0, x_2 = 1, x_3 = 2, x_4 = 3, x_5 = 3$ and $x_6 = 4$ for the Base and distinct value networks. . . . .	42
3.3	Convergence of the social network under the policy of distinct value and considering $n = 6$ and $\eta = \frac{n}{2}$ where the ellipses represent a cluster of nodes with equal opinions. . . . .	51
3.4	Convergence of the social network under the policy of circular value and considering $n = 5$ and $\eta = 1$ , where the ellipses represent a cluster of nodes with equal opinions. . . . .	53
3.5	Convergence of the social network under the policy of Distinct Neighbor and considering $n = 6$ and $\eta = 1$ , where the ellipses represent a cluster of nodes with equal opinions. . . . .	54
3.6	Evolution of $V(k)$ for the case of a base social network for values of $\eta = 16, \dots, 20$ . . . . .	63
3.7	Evolution of $V(k)$ for the case of a social network with agents communicating with nodes with distinct opinions for values of $\eta = 8, \dots, 12$ . . . . .	64
3.8	Evolution of $V(k)$ for the case of a social network with agents with strong opinion looking for opposite opinions for values of $\eta = 1, \dots, 5$ . . . . .	64
3.9	Evolution of $V(k)$ for the case of a social network with agents contacting the $2\eta$ closest distinct neighbors for values of $\eta = 1, \dots, 5$ . . . . .	65
3.10	Comparison of the evolution of $V(k)$ for the four cases with $\eta = 1$ . . . . .	65
3.11	Evolution of the final state $x_\infty$ as function of $\eta$ for the case of the base network dynamics. . . . .	67
3.12	Evolution of the final state $x_\infty$ as function of $\eta$ for the case of the Distinct Network dynamics. . . . .	67
3.13	Evolution of the final state $x_\infty$ as function of $\eta$ for the case of the Neighbor Network dynamics. . . . .	68
3.14	Evolution of the final state $x_\infty$ as function of $\eta$ for the case of the Circular Network dynamics. . . . .	69
4.1	Example of the sets produced by the SVOs. . . . .	81

## List of Figures

---

4.2	Example of the set-valued estimates boundaries of node $i$ (yellow), node $j$ (green) and node $\ell$ (red), where for each node there is no uncertainty regarding its own state and where $s^*$ represents the full state of the system that is contained in all three state boundaries. . . . .	90
4.3	Flowchart of the algorithm with the intersection phase to share observations between neighbors. . . . .	91
4.4	Illustrative example of the setup for the problem. . . . .	101
4.5	Communication graph used for simulation. . . . .	106
4.6	Detection times for the stochastic fault. . . . .	107
4.7	Detection times for the deterministic fault. . . . .	108
4.8	Detection times for the SSVO. . . . .	109
4.9	Average difference between detecting with a SVO in one node or in all the nodes. . . . .	110
4.10	Detection time for different horizon values for a fault constant equal to 3. . . . .	110
4.11	Detection time for different horizon values for a fault constant equal to 4.9. . . . .	110
4.12	Detection time for different fault constants. . . . .	111
4.13	Typical behavior of the size of the SVO. . . . .	112
4.14	Histogram for the stopping time with the proposed algorithm. . . . .	113
4.15	Evolution of the mean sum of edges of all node set-valued state estimations. . . . .	113
4.16	Final distribution of the nodes after 100 time instants using one antenna. . . . .	114
4.17	Evolution of the maximum distance between two nodes over the 100 time instants of the simulation using one antenna. . . . .	115
4.18	Final distribution of the nodes after 100 time instants using both antennae. . . . .	115
4.19	Evolution of the maximum distance between two nodes over the 100 time instants of the simulation using both antennae. . . . .	116
5.1	Schematic representation of the two coprime systems. . . . .	125
5.2	Illustration of the fault detection mechanism resorting to the intersection of the sets generated by the SVOs of each subsystem resulting from the coprime factorization. . . . .	128
5.3	Output of the mass-spring-dashpot system with a fault introduced after 4 seconds. . . . .	132
5.4	Detection time as a function of the magnitude of a constant fault introduced after 4 seconds. . . . .	133
5.5	Mean detection time as a function of the magnitude of a random fault introduced after 4 seconds. . . . .	134
5.6	Detection time as a function of the magnitude of a sinusoid fault introduced after 4 seconds. . . . .	134
5.7	Detection time as a function of the magnitude of a alternating fault introduced after 4 seconds. . . . .	135

---

6.1	Schematic representation of the two coprime systems. . . . .	144
6.2	Flowchart of an iteration of the Fast SVO algorithm which takes as input the coprime factorization and decides if the model is invalid or still valid. . . . .	148
6.3	Example of a simple fault detection where the state of the system (blue line) crosses the upperbound (red line) of the state given through the projection of the set-valued estimate onto the corresponding coordinate. . . . .	150
6.4	Reported detection times when varying the magnitude of a constant fault. . . . .	151
6.5	Mean detection times when varying the maximum magnitude of a random fault. . . . .	152
6.6	Reported detection time for a fault free system but with unmodeled disturbances. . . . .	152
6.7	Detection and isolation of fault $f_1$ in the system. . . . .	153
6.8	Lower and upper bounds of the set-valued estimates when not in the presence of disturbances. . . . .	154
6.9	Hypervolume of the set corresponding to the system $S_G$ for eigenvalues of $A-KC$ close to zero (deadbeat) and with $\lambda_{\max} = 0.74$ . . . . .	154
6.10	Hypervolume of the set corresponding to the system $S_G$ for eigenvalues of $A-KC$ close to zero (deadbeat) and uncertainty of 1 and $10^6$ for the initial state. . . . .	155
6.11	Running time of the SVOs compared with the fSVOs. . . . .	156
6.12	IEEE 14 bus system test bed example [oW15] . . . . .	156
6.13	SVO tracking of the true state of node 1 in the network. . . . .	157
7.1	Block diagram of a NCS. The Event Detector and Event Scheduler blocks implement event- and self-triggered strategies, respectively, based on the set denoted by $X(k)$ produced by the observer. . . . .	163
7.2	Original and rotated sets, blue and green respectively, and its correspondent overbounds. . . . .	166
7.3	Counterexample where a set is rotated but a worst overbound is achieved. . . . .	169
7.4	Example of the evolution of Algorithm 5 for a polytope that is not centered and not centrally symmetric. Edges are counted starting at the top one and counterclockwise. . . . .	170
7.5	Abstract example where the previous set $X(1)$ is enveloped by the hyper-parallelepiped approximation in dashed line and the ellipse upper bound. . . . .	172
7.6	Example of using SVOs for self-triggered systems. At time $\tau^{-1}(k)$ , the observer computes set $\tilde{X}(\tau^{-1}(k))$ and propagates twice to get $\tilde{X}_p(k)$ and $\tilde{X}_p(\tau^1(k))$ , which is larger than $\tilde{X}(\tau^{-2}(k))$ , and triggers a sensor measurement, making the intersection with the measurement set $Y(\tau^1(k))$ to get the new estimation $\tilde{X}(\tau^1(k))$ . . . . .	175
7.7	Original set and ellipsoidal overbound with the set resulting from the intersection with the measurement set to form the new set-valued estimate. . . . .	177

## List of Figures

---

7.8	Flowchart of the Self-Trigger SVO algorithm where $\mathcal{E}(k)$ and $\mathcal{B}_{\mu(k)}$ are the over-bounding ellipsoid at time $k$ and the ball of radius $\mu(k)$ centered at the origin, respectively. . . . .	178
7.9	Depiction of the observer and sensor sets for a combination of a Self-Triggered SVO used with an event-triggered NCS. . . . .	180
7.10	Network example for a distributed system. . . . .	180
7.11	Example demonstrating two ellipsoids and its corresponding intersection with the set of observations. . . . .	182
7.12	Estimation conservatism and triggering frequency of the event-triggering strategy for NCS using the standard SVOs. . . . .	189
7.13	Estimation conservatism and triggering frequency of the self-triggering strategy for NCS using the standard SVOs. . . . .	191
7.14	Estimation conservatism and triggering frequency of the Self-triggered SVOs in comparison with the standard SVOs. . . . .	192
7.15	Elapsed time in seconds of the computation of the estimates using the standard and Self-triggered SVOs. . . . .	193

# LIST OF TABLES

- 2.1 Second largest eigenvalue for the bidirectional ( $b_{\lambda_2}$ ) and the presented unidirectional ( $u_{\lambda_2}$ ) algorithms for the 3 studied cases and for the Expectation and Second Moment. . . . . 30
- 2.2 Upper and lower bounds for the mean square on the number of ticks for the algorithms to reach in a neighborhood of the solution of  $\epsilon = 10^{-2}$  for the bidirectional case ( $b_{\text{ticks}}$ ) and the presented unidirectional ( $u_{\text{ticks}}$ ) algorithms. . . . . 31





2.1	Definition (Stochastic Convergence) . . . . .	15
2.1	Theorem . . . . .	17
2.2	Theorem (Convergence of $\mathcal{G}$ ) . . . . .	18
2.2	Definition (disagreement) . . . . .	24
2.3	Definition (nonexpansive and pseudocontraction) . . . . .	24
2.4	Definition . . . . .	24
2.1	Lemma . . . . .	24
2.3	Theorem (Convergence of $\mathcal{B}$ ) . . . . .	25
2.5	Definition ( $\epsilon$ -averaging time) . . . . .	27
2.4	Theorem (Convergence in discrete time) . . . . .	27
2.5	Theorem (Distributed Optimization) . . . . .	28
3.1	Definition (order of) . . . . .	39
3.2	Definition (base network) . . . . .	40
3.3	Definition (distinct value) . . . . .	40
3.4	Definition (distinct neighbors) . . . . .	41
3.5	Definition (circular value) . . . . .	41
3.6	Definition . . . . .	43
3.1	Lemma (order preservation) . . . . .	44
3.2	Lemma (convergence for higher connectivity) . . . . .	45
3.1	Theorem . . . . .	47
3.1	Remark (Distinct state values) . . . . .	47
3.1	Proposition . . . . .	48
3.2	Theorem (Base Network Final Opinion) . . . . .	48
3.2	Remark (symmetric case) . . . . .	49
3.2	Proposition . . . . .	49
3.3	Theorem . . . . .	50
3.4	Theorem . . . . .	51
3.5	Theorem . . . . .	52
3.3	Remark . . . . .	53
3.6	Theorem . . . . .	53
3.7	Theorem . . . . .	54
3.8	Theorem . . . . .	56
3.9	Theorem . . . . .	59
3.10	Theorem . . . . .	60

## List of Theorems

---

4.1	Definition (undetectable faults) . . . . .	76
4.1	Assumption (bounded state) . . . . .	78
4.2	Definition (Fourier-Motzkin elimination method [Tel82]) . . . . .	78
4.1	Proposition ( $X(k+1)$ computation [ST99]) . . . . .	79
4.2	Proposition (Growth of $\tilde{X}(k)$ ) . . . . .	82
4.3	Definition ( $N_d^\star$ ) . . . . .	83
4.3	Proposition (SVO with local information) . . . . .	83
4.1	Theorem . . . . .	86
4.1	Remark (Bound in the Horizon) . . . . .	86
4.4	Definition ( $\alpha$ -confidence sets) . . . . .	87
1	Property . . . . .	88
4.5	Definition . . . . .	92
4.2	Theorem . . . . .	92
4.4	Proposition (Attacker signal bound) . . . . .	95
4.1	Corollary (Attacker signal bound for SSVO) . . . . .	97
4.3	Theorem . . . . .	98
4.4	Theorem . . . . .	99
4.2	Remark . . . . .	100
4.5	Theorem . . . . .	105
1	Problem (Fault Detection) . . . . .	121
5.1	Definition (Uniformly $n_x$ -step Observable [Lev96]) . . . . .	121
5.1	Assumption . . . . .	122
5.2	Definition (coprime factorizations [RPK92]) . . . . .	125
5.1	Proposition . . . . .	126
5.1	Theorem . . . . .	127
5.2	Proposition . . . . .	131
6.1	Lemma (fault detection) . . . . .	143
6.1	Proposition (left-coprime factorization [ZDG96]) . . . . .	144
6.1	Definition . . . . .	144
6.1	Theorem (estimate convergence) . . . . .	145
6.2	Definition (fault detectability [PDB11]) . . . . .	147
6.3	Definition (fault distinguishability) . . . . .	147
6.2	Lemma (fault detection) . . . . .	148
2	Problem (Triggering in the worst-case) . . . . .	164
3	Problem (Triggering with stochastic information) . . . . .	164
7.1	Proposition . . . . .	166
7.2	Proposition . . . . .	167

---

7.1	Definition . . . . .	168
7.1	Theorem . . . . .	170
7.1	Corollary . . . . .	171
7.1	Remark . . . . .	171
7.2	Definition (reordering property) . . . . .	180
7.2	Remark . . . . .	181
7.3	Proposition . . . . .	181
7.2	Theorem . . . . .	181
7.4	Proposition . . . . .	183
7.3	Theorem (SVO convergence) . . . . .	183
7.4	Theorem . . . . .	184
7.3	Definition (volume expansion stochastic variable) . . . . .	185
7.4	Definition (upcrossing) . . . . .	186
7.5	Theorem . . . . .	186



# 1

## INTRODUCTION

### 1.1 Motivation

Current technical and theoretical developments in systems, electrical and network engineering are making possible the implementation of distributed networked systems. As a consequence, future networked control systems are going to have both a human interaction as well as a physical system component and be the support basis to a wide range of applications with high scientific and commercial added value. We have entered in an era where common-use devices are ubiquitous and have resources and a number of embedded processors rapidly outrunning those in traditional computers. People are adapting and getting familiarized with cities, rooms, robots and other otherwise ordinary objects being enhanced with computational capabilities to better perform their purposes. Human understanding of its surrounding environment and how efficiently the available resources are used in a sustainable way, safeguarding critical structures and operations, and, addressing the challenges posed to mankind have everything to benefit from the availability of miniaturized sensors and actuators, embedded processors and wide-coverage communications networks. The research and industrial communities play a key role in this process by supplying the required methods for information processing, machine learning, systems optimization, computer vision, decision-making and control.

Networked systems present challenging tasks such as distributed decision making and control of complex and heterogeneous structures, distributed energy management, optimal sensor placement for monitoring potential hazardous areas (such as erupting volcanoes) and analysis of high frequency trading. Huge amounts of generated data make imperative the adoption of new processing tools based on advanced distributed inference methods, information retrieval in large databases and data classification. Systems must be capable to negotiate strategies, devise techniques to exchange data, assess the well-functioning of the remaining components, and achieve goals cooperatively while satisfying strict energy and resource allocation and communication constraints.

The research problems in Networked Control Systems (NCSs) and distributed systems have stringent constraints in the type of adopted solutions, in particular, requiring preferably non-existing centralized tasks as to avoid unnecessary overhead in the communications and reduce the performance of the network. Tackling the problems of incorporating the stochastic behavior of nodes; removing structure of the algorithm in order to make it more robust; and considering all possible communication patterns when detecting faults can render decision systems computationally intractable if the algorithms do not scale properly with the size of the state space and the number of possible interactions among nodes.

The aim of this thesis is to address the above mentioned issues in the design of observers and decision systems for NCSs, incorporating the stochastic behavior in the fault detection procedure and by developing the necessary tools suited to reduce the computational and network cost. The target will be to design low-complexity distributed observers while maintaining the accuracy by exploiting the network structure in their algorithms. In doing so, it is possible to discard irrelevant information and dynamics that would otherwise increase the computational load without significant gains in the accuracy of the estimation/decision. A reduction is also made possible by rewriting the equations defining the current state or the occurrence of a fault. Triggering techniques also play a key role in adapting the proposed algorithms to be applied to NCSs. In the context of NCS, estimating the state or detecting faults with observers running in the NCS loop over the network is a complex task due to the necessity of having distributed solutions involving a large number of states, varying network topology, and possible interactions among nodes. In this thesis, research focus is given to three application scenarios:

**Consensus networks** nodes spread ubiquitously over an area, measure or acquire quantities of interest and transmit them to their neighbors to reach an agreement over the initial values. These networks are characterized by having a structure created in an ad-hoc fashion, which can vary over time due to nodes switching off and to the stochastic characteristics of the communication. Network topology is assumed to evolve independently from the state of the nodes. Such an assumption is hard to remove as most tools to prove stability assume independent communications. The convergence and correctness of the algorithm highly depend on the absence of faults since they are synchronous;

**Social networks** modeling how people interact and reach conclusions can benefit other practical cases, such as those where nodes use a wireless medium and have to make decisions regarding their position. The main objective is to consider a state-dependent evolution of the network that presents convergence properties of interest. The analysis of state-dependent networks require a distinct approach from those used in consensus with the assumption that, even though the topology changes over time, those changes are independent from the state of the system;

**Smart grids** with research efforts and companies investing in modernizing the electrical

power grids towards creating smart grids (i.e. energy networks that can automatically monitor energy flows and adjust to changes in energy supply and demand accordingly), an important aspect in ensuring its continuous operation is the detection of malfunctioning components, outages in power sources, load buses that fail, communications errors between appliances, etc. that can perturb the overall power grid performance. According to the GE company website, “Power Interruptions cost European Union businesses €150 billion each year. Outages cost the U.S. economy an average of \$1.5 billion each week - \$80 billion, with a ‘B’ each year.”. One important problem in designing observers and decision making mechanisms for these kind of networks is related to the fact that the observability of the whole system can be compromised by the presence of similar components, i.e., components with the same dynamics. In the case that only relative measurements are available (i.e., the difference between each pair of states), observability is lost. It motivates to consider how to design distributed tools for fault detection and isolation that can deal with the above problem without compromising the required accuracy. The solution should be distributed for fault detection and isolation with multiple detectors, thus potentially reducing the time to detect faults and the rate of missed detection.

## 1.2 Previous Work and Brief Literature Review

State-of-the-art techniques to estimate the state in the context of NCSs applications use filters to obtain estimates of the state as well as bounds on the error for those estimates. On the other hand, considering the worst-case scenario entails the use of techniques like set-valued estimators. One trend is to rely on the concept of zonotopes, described in [BR71] and further developed in [Com05] and [ABC05]. Other alternatives use polytopes, such as Set-Valued Observers (SVOs) introduced in [Wit68] and [Sch68] and further information can be found in [Sch73] and [MV91] and the references therein. If the algorithm performs stochastic decisions or in the context of randomized distributed systems, then it is not fully addressed how to perform set-valued state estimation and fault detection taking into account the stochastic characteristics of the information available in the proposed target applications.

Algorithms designed for NCSs and for distributed networks often resort to dynamics that are state-dependent either due to the interaction with people or because of the conditional rules in their definition. In this context, even proving convergence of simple distributed linear iterative processes such as consensus (see, e.g., [OSM04], [BCM09], [HSJ14], [CHT14] and [DGH13]) relies mostly in tools assuming no state-dependence. Convergence results for general stochastic systems with independent selection of dynamics were given in [TN14]. Many other topics have attracted research interest such as: the study of stochastic packet drops and link failures in [PBEA10], the existence of delays in [HC14] and [FZ09], quantized data transmissions in [CBZ10], state-dependent noise in [LWZ14] and time-varying communication connectivity [OSM04] [CI14]. Nevertheless, the mentioned analysis tools are not suitable to deal with

dynamical systems with state-dependent rules.

In the context of distributed systems and networked control systems, the performance bottleneck is often in the communication network either because a large number of nodes compete for access to the shared medium or the network available bandwidth is less than that required by the control loops. For example, if there are several processes to be controlled, the controller and the sensors might be in different spatial locations and compete for network access. The problem is highlighted if either the state space or the number of NCSs using the same communication infrastructure is of large dimension.

In the control community, two main strategies have emerged to reduce the communication rate in discrete time closed-loop systems, namely: event-triggered, where the sensor decides based on the current measurements if it should transmit to the controller/observer; and self-triggered where the controller/observer decides, based on the available information, i.e. the current estimate of the state, when the sensor should perform the next update. An event-triggered solution results in a more informed choice since the sensor has access to the actual measurement; however, it prevents the sensor to be shut down between updates. For a recent discussion of event- and self-triggered control and estimation please refer to [HJT12].

The strategy for an observer to self-trigger a sensor update based on its estimates can resort to an optimization over the update patterns such as in [AH14], where the disturbances and noise are assumed to be Gaussian distributed. In [ASP14], an estimator of the type of Kalman filter is proposed for state estimation, which lacks the computation of an error bound. For event-triggered systems, the condition can be on the norm of the estimation error being below a given threshold factor of the norm of the state [MT11]; requiring the derivative of a Lyapunov function of the state being semi-negative definite [MT08] [HJT12]; or, having the norm of the state below a certain threshold [HSB08]. However, more general event- and self-triggered strategies need to be developed for networked control systems with the objective of having an online strategy, meaning that, at each time instant, the observer or the sensor must be capable of deciding when the next measurement update is going to take place. Those techniques must be distributed, able to handle large dimension state spaces and number of sensors as well as being suitable to produce set-valued state estimates.

A trade-off between accuracy and computational cost is common when designing set-valued state estimators. In order to increase the accuracy of the observers, one needs to consider more past instant observations as to reduce the initial uncertainty of the state, thus increasing the computational complexity and rendering the solution not suitable for time sensitive applications or cases where nodes have limited computational resources. The problem of reducing the complexity of the SVOs simultaneously improving the respective convergence for Linear Time-Invariant (LTI) systems resorts to using a left-coprime factorization [RSA14].



## 1.3 Contributions of the Thesis

The main contributions of this PhD thesis are as follows:

- The design of randomized gossip and broadcast algorithms to solve the average consensus problem that is able to cope with crash-type faults in the network such as packet loss and nodes entering and leaving the network. Convergence rates are provided both for the continuous and discrete time case given by the relationship between the two. Convergence is shown for three important stochastic definitions and then provided how the problem of optimizing the algorithm parameters can be carried in a distributed fashion;
- A model for social interactions based on proximity of objective opinions describing how people interact and reach conclusions is proposed, resulting in a linear state-dependent algorithm with finite-time convergence properties. The issue of determining which nodes contribute the most to the final opinion is addressed and the randomized version of the social network is introduced with a two-fold objective: model more accurately the asynchronous behavior of social interactions, and allowing to implement the same algorithm for problems with wireless communication where the network is dependent on the position of the nodes;
- The concept of Stochastic Set-Valued Observers (SSVOs) is described as a mathematical tool to find set-valued estimates of the state representing  $\alpha$ -confidence sets for distributed systems. Detection of faults representing a different model for the probability distributions is made possible using SSVOs as well as results regarding the maximum input of third party in the distributed system before being detected;
- A finite-time detection algorithm with nodes sharing measurements is constructed using SVOs or SSVOs and an analysis is made as to provide results regarding the maximum horizon value needed and the links of the topology graph that are irrelevant for the case of distributed algorithms;
- An extension to existing results bounding the horizon value by the size of the state space for LTI systems is given for the more general class of Linear Parameter-Varying (LPV) systems by resorting to a left-coprime factorization and the definition of deadbeat observers;
- In the context of power networks, it is shown that by performing a coprime factorization, the SVOs can be applied to detectable plants, where the convergence of estimates is governed by the slowest stable unobservable mode;
- and, event- and self-triggering strategies are presented using SVOs for NCSs. Additionally, using this concept, it is described the procedure to reduce the computational cost at the expenses of temporarily increasing the conservatism of the estimates which is fundamental when considering time-sensitive plants or real-time applications.

Each of the following chapters contains a list of its specific contributions.

### 1.4 Organization of the Thesis

#### 1.4.1 Randomized Time-Dependent Algorithms

Even assuming that all conditions for convergence of synchronous or asynchronous algorithms are met, in the general case, faults can drive the system to final states that do not correspond to the desired operating point, and in worst cases, convergence can even be prevented. Moreover, if we assume the fault is being caused by an external agent that is trying to compromise the system, without any type of fault detection mechanism or fault-tolerant algorithms, simple malicious actions can drive the state of the system to wherever him/she desires depending only if the nodes that can be corrupted form a controllable system for the attacker.

Therefore, Chapter 2 addresses the design of randomized gossip and broadcast algorithms that are robust to crash-type faults in the network, such as unresponsive nodes, packet drops, packets discarded by failed checksum values, etc. The average consensus problem is studied due to its many interesting applications and connections to other problems in the control community. The adopted model assumes that the behavior of the network is independent which precludes the use of many available tools in the literature to prove convergence and give expressions for the rates of convergence of the algorithms.

Special interest is focused on the convergence rate for the proposed algorithms. To this extent, the continuous case convergence rate is expressed at the expenses of the discrete case. Finding the fastest convergence algorithm relates to solving an optimization problem that, through an explicit relationship between the probability distribution of communications and the convergence rate, it is possible to write as a convex optimization problem and employ standard strategies in the literature and obtain a distributed optimization of the convergence rate.

#### 1.4.2 Randomized State-Dependent Algorithms

A crucial assumption in Chapter 2 is the independent evolution of the network. The nodes state plays no role in the definition of neighbors or actions to be followed. Performance is limited by this fact and, in many cases of interest where nodes are connected through a wireless medium, the network topology might depend, for example, on the positions of the nodes.

Chapter 3 tackles the problem of understanding how people reach conclusions from their initial objective opinions regarding a subject. The main motivation behind looking at this problem is that by removing the aforementioned assumption, performance can be improved or, conversely, the demand for network resources can be relaxed.

The stochastic version of the social network results in an asynchronous distributed algorithm that better maps how people interact but also has interesting robustness properties to crash-type

faults if applied to control problems. Study of convergence requires different techniques as the assumption for independent network formation no longer applies.

### 1.4.3 Set-Valued Estimators

In Chapter 2 and Chapter 3, the class of faults being tolerated by the algorithms was limited to crash-type faults. However, many other faults can affect a plant with faulty sensors or actuators having the same impact on the performance as crash faults have in synchronous algorithms. Chapter 4 considers a broader class of faults and more realistic models by allowing parameter uncertainties, disturbances and noise in the sensors.

Progress in the construction of fault-tolerant systems is made by firstly defining the problem, which motivated the use of set-valued estimators due to their guarantees for the worst-case scenario. In a sense, if the faults are caused by an agent with malicious intentions, then even if the probability of a certain event is very small but with a big impact, then it must be considered. The framework for SVOs is introduced allowing the computation of a polytopic set where the state is guaranteed to belong. As a consequence, a bound on the attacker signal can be computed such that, if it is exceeded, the fault is guaranteed to be detected.

In a different direction, a fault in a stochastic system can also be the consequence of certain events happening with a probability distribution different from the assigned by the algorithm. To deal with those faults, the concept of Stochastic Set-Valued Observers (SSVO) is proposed to compute  $\alpha$ -confidence sets where the state is guaranteed to belong with probability  $1 - \alpha$ .

The discussion converges to the proposal of a randomized algorithm for the consensus problem where nodes share estimates and that achieves finite-time detection in case of a fault. Even if the sequence of transmissions is not rich enough, at least the algorithm converges asymptotically to the solution. In practice, this algorithm can be enforced to satisfy the finite-time property by employing a token-passing scheme. Depending on the type of set-valued estimators to be used, different classes of faults can be detected and isolated by the algorithm.

### 1.4.4 SVOs for LPV systems with Coprime Factorization

The main disadvantage concerning the use of SVOs is its associated high computational cost that grows exponentially with the number of uncertainties and the horizon. In Chapter 4, results specific for distributed systems having in mind the application of consensus were presented both to reduce the number of links considered in the model (i.e., also reducing the number of required uncertainties) and the number of past observations (i.e., the horizon). However, the structure of the algorithm and some properties regarding its dynamics were used which invalidates the same analysis for the general case.

Chapter 5 is dedicated to the problem of bounding the required horizon to guarantee that the size of the sets representing the estimates does not grow without bounds. By performing a left-coprime factorization, the original system is divided into two stable subsystems with

dynamics that can be made as fast convergent as desired if there exists a deadbeat observer for the system. Thus, by constructing an SVO for each of the subsystems it is possible to eliminate the error associated with past estimates and the uncertainty in the initial state with a horizon equal to the size of the state space. It also makes possible fault detection for unstable plants, which was not guaranteed for the original plant as the SVOs can introduce conservatism if approximated methods are used to save in complexity.

### 1.4.5 Fault Detection and Isolation in Detectable Systems

For some applications in networks and distributed detection, observability might be lost for instance when only relative measurements are available. In such cases, even though the system is normally stable, the fact that it has unobservable modes means that the set-valued estimates are going to diverge. In order to apply the techniques described in this thesis to such cases, the issue of detectable systems must be addressed.

For detectable systems, Chapter 6 revisits the topic of using a coprime factorization and where the modes of the dynamics of the observer can be placed arbitrarily except for the unobservable modes. In doing so, the convergence rate for the estimates is going to depend on the slowest unobservable mode. The discussion regarding the convergence led to consider the definition for fault detectability and fault identifiability in the literature as to define new equations that do not require a projection using the Fourier-Motzkin elimination method and, therefore, are of low complexity when compared to the standard SVOs. The new type of SVOs are not iterative by nature which means that conservatism is added to the initial estimate but is removed by the fast dynamics of the observer using the factorization.

### 1.4.6 Event- and Self-Triggered NCS and Set-Valued Observers

Chapter 5 initiated a discussion regarding the speed of the computations of the set-valued estimates. Real-time applications or plants to be discretized using a small sampling period demand fast observers. The proposal in Chapter 7 is to overbound the sets produced by the SVOs using ellipsoids and propagate them by resorting to the techniques used for set-valued estimators with ellipsoids. In doing so, we have a low-complexity approximation at the expenses of adding conservatism. Event- and Self-triggering strategies are introduced to cope with the problem that the added conservatism can deteriorate the accuracy above a certain threshold or above a certain level where the estimates are not converging.

In studying the aforementioned strategies to reduce the number of times the full iteration of the SVOs is computed made natural the application of similar conditions for event- and self-triggering NCS. In this setup, the main objective is to reduce the number of times the sensor communicates its measurement to avoid consuming the resources of the network. The triggering frequency is studied both in the cases where no information is known regarding the parameters of the dynamics and when the stochastic distribution has a known expected value

for the maximum singular value of the dynamics.

## 1.5 Notations and Definitions

This section introduces some of the mathematical notation used throughout the thesis. Further details will be presented later as necessary. Specific definitions to each chapter are introduced in that same chapter for clarity.

$\mathbb{R}^n$	set of ordered $n$ -tuples of real numbers,
$\mathbb{R}^{n \times m}$	set of $n$ by $m$ matrices with elements in $\mathbb{R}$ ,
$x^\top$	transpose of a vector $x$
$A^\top$	transpose of a matrix $A$
$r_\sigma(A)$	spectral radius of matrix $A$ ,
$\lambda_i(A)$	$i$ -th eigenvalue of $A$ ,
$\sigma_i(A)$	$i$ -th singular value of $A$ ,
$\sigma_{\max}(A)$	maximum singular value of $A$ ,
$\mathbf{1}_n$	$n$ -dimension vector of ones,
$\mathbf{0}_n$	$n$ -dimension vector of zeros,
$\mathbf{e}_i$	vector of zeros except the $i$ th entry which is equal to 1,
$[x]_i$	$i$ th component of vector $x$ ,
$I_n$	identity matrix of dimension $n$ ,
$\text{diag}([A_1 \dots A_n])$	block diagonal matrix with blocks $A_i$ ,
$\otimes$	Kronecker product,
$O(n)$	Orthogonal group of dimension $n$ .

For a vector  $v \in \mathbb{R}^n$ , we define the vector norm of  $v$  as

$$\|v\|_p := \left( \sum_{i=1}^n |v_i|^p \right)^{\frac{1}{p}},$$

for  $1 \leq p \leq \infty$ . The subscript  $p$  is dropped whenever clear from context that we are referring to  $p = 2$ . For a matrix  $A \in \mathbb{R}^{n \times n}$ , we define the matrix norm induced by the above vector norm as

$$\|A\| := \sigma_{\max}(A).$$

For a matrix  $A \in \mathbb{R}^{m \times n}$ ,

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

the operation  $\text{vec}(A)$  returns a  $mn \times 1$  column vector corresponding to stacking all columns of matrix  $A$ , i.e.,

$$\text{vec}(A) := [a_{11}, \dots, a_{m1}, a_{12}, \dots, a_{m2}, \dots, a_{1n}, \dots, a_{mn}]^\top.$$



# 2

## RANDOMIZED TIME-DEPENDENT ALGORITHMS

### 2.1 Introduction

Distributed iterative algorithms performance and correctness can be prevented when in the presence of packet drops, transmission errors, reordering of the messages in the communication channel, among many other issues that might arise from considering a network connecting the different parts in the loop between controller and plant or between agents in a multi-agent systems scenario. In order to make protocols robust to these issues, a possibility is to address a classical point-of-view and define thresholds for considering a packet lost, maintaining counters to ensure ordering of the messages, etc. However, modern medium access technologies such those used for wireless communication exploit randomness to remove the necessity for highly-structured algorithms and better usage of the medium.

This chapter follows the concept of introducing random transmissions and decisions to deal with those issues, while still providing results that can give performance guarantees in a stochastic sense. The case of time-dependent interaction with no relationship with the agent states is addressed and events can be considered to be independent, with traditional tools for analyzing Markov Chains and convergence of stochastic variables being employed. The particular case of consensus is quite common in the literature but usually prone to having its correctness and performance compromised in the presence of faults.

### 2.2 Main Contributions and Organization

This chapter is organized as follows. The case of consensus is presented in a setting tolerant to any type of fault regarding the network by focusing on designing an asynchronous algorithm with directed communication in such a way that only a node receives information and therefore no additional communication is required to ensure that the transmission is successful. In essence, the node is going to change its state according to the algorithm iteration if and only if it

received a packet, without requiring any type of ordering or guarantees. For this case, results about the convergence rate and how to optimize its performance are provided in a distributed manner.

The main contributions of this part are two-folded and were presented in the papers [ASS11] and [SRHSew]:

- we introduce a new algorithm based on state augmentation to deal with the case that communication is unidirectional in each time slot. We consider two scenarios, namely the *gossip* — where each node communicates with one neighbor; and *broadcast* — where each node transmits to the whole network but does not receive information at that time slot. We show convergence for three different stochastic convergence definitions and present necessary and sufficient conditions for convergence. Results regarding convergence rates in discrete time are presented for both scenarios;
- we address the problem of finding the fastest convergent directed algorithm showing how it can be written as a Semi-definite optimization problem and how nodes can solve it in a distributed fashion.

### 2.3 Consensus

Consensus refers to the problem where a group of agents needs to agree on a function of their initial state by means of a distributed algorithm, in which the communication between agents is constrained by a network topology. Such problem is of prime importance and examples of application range from distributed optimization [TBA86], [JKJJ08]; motion coordination tasks like flocking, leader following [JLM03]; rendezvous problems [CMB06]; and resource allocation in computer networks [CLCD07].

The average consensus problem has been solved using linear distributed algorithms with each agent computing a weighted average of its state and the values received from its neighbors (see, e.g., [OSM04], [BCM09]). Several instances of this problem have been proposed such as considering stochastic packet drops and link failures [PBEA10], [FZ09], quantized data transmissions [CBZ10], and time-varying communication connectivity [OSM04].

The above variations of the consensus problem are prone to faults affecting its performance and correctness. Since nodes states evolve deterministically and synchronously, it means that average is not kept if one communication fails, which entitles for the need to have every node determining if the whole interaction was successful before committing to the updated value. An important class of solutions capable of dealing with a varying network topology caused by nodes joining and leaving the network was introduced in [BGPS06] as a randomized gossip algorithm. The main feature of this algorithm is that each agent communicates with a randomly selected neighbor at each transmission time. In [BGPS06], pairs of nodes exchange their state information, which assumes bidirectionality in the communication.



In this chapter, it is presented a generalization to the unidirectional case and considered the case where a node can broadcast to the entire network, which appears naturally in wireless networks. In doing so, the need to consider faults for each link is removed as only the node receiving information changes its state. Communication is unidirectional at each time slot, i.e., at each transmission time a single agent transmits data to one or several agents, but does not receive data. This is of interest to construct algorithms tolerant to packets being discarded or lost. Note that at a different time slot receiver and sender agents may invert their roles, i.e., the word unidirectional refers only to communication at a given transmission time.

We consider the two following scenarios: (i) randomized gossip algorithms in wireless networks, where each agent becomes active at randomly chosen times, transmitting its data to a single neighbor; (ii) broadcast wireless networks, where each agent transmits to all the other agents, access to the network occurs with the same probability for every agent, and the intervals between transmissions are independent and identically distributed. As we shall see, the unidirectionality communication constraint precludes in general the existence of a linear distributed algorithm where associated to each agent there is a single scalar state. The state of a node is updated based on the values of the other agents, as in related problems where the communication topology of the network is also time-varying, but satisfies different assumptions (see [OSM04], [BCM09]). We assume a symmetric communication topology, meaning that if an agent  $a$  can communicate with an agent  $b$  then the agent  $b$  can communicate with the agent  $a$ , although this does not take place at the same transmission time, i.e., at each transmission time the graphs modeling communications are in general asymmetric. Note that this is typically the case in wireless networks, and therefore this assumption is reasonable to assume in both scenarios (i) and (ii).

Directly related to our study of the fastest distributed algorithm is [BGPS06] and [CI12]. As *supra* cited, [BGPS06] considers bidirectional communications but provides upper and lower bounds on the convergence to the average consensus. More recently, [CI12] proposes a linear algorithm that almost surely converges to consensus and also provides convergence rates. Nonetheless, the algorithm assumes that at each transmission time a node communicates with all its neighbors, instead of a single one. In [LM12], a technique using a scaling variable is employed and the network model consists of all nodes communicating to its neighbors with the correspondent communication graph being strongly connected. In [CI11], a gossip algorithm is presented using asynchronous communication between the pairs of nodes. The average consensus is achieved using a state augmentation technique and a nonlinear operation based on the received state and the node's own state. The method does not assume a symmetric communication topology, but it is only proved to converge almost surely and not in mean square sense. Our algorithm is the directed linear parallel of the standard gossip algorithm presented in [BGPS06] and relates to the linear distributed algorithms [BCM09].

The study of convergence using ergodic infinite sequences of stochastic matrices has also

been applied to study the consensus problem. In [TSJ08], the underlying network is generated by a random graph process and convergence is shown to be equivalent to the spectral radius of the expected value matrix having the second largest eigenvalue inside the unit circle. The chain product of stochastic matrices is studied in [TN14] for balanced and strongly aperiodic chains. In [BBT<sup>+</sup>10], the concept of ergodicity is explored to prove that a weighted gossip algorithm, which uses a variable to estimate the sum of all initial states and a weight variable to count the number of nodes, converges to the average consensus. These proposals using the ergodicity concept require each matrix in the chain to have strict positive diagonal which differs from the class of algorithms studied in this chapter. The same concept of a variable to track the sum and another for the number of nodes is used in [KDG03], even though, the main focus is on bounds for the time of convergence. In [DGHe], multiple dynamic weight assignment techniques are proposed and the algorithm is showed to converge if the underlying graph is strongly connected. In essence, all these proposals that require strongly connected graphs as the support graph for each update matrix differ from our work in the sense that in each iteration more than a pair of nodes needs to communicate.

These results can be found in a preliminary version in [ASS11] and extended by providing a proof for converge in mean square sense and almost surely and also by showing how to use the structure of the expected value matrix to simplify the nonconvex optimization of the average of the non-symmetric transmission matrices.

### 2.3.1 Problem Description

We consider a set of  $n$  agents with scalar state  $x_i(k)$ ,  $1 \leq i \leq n$ , and our goal is to construct a distributed iterative algorithm that guarantees convergence of the state to its initial average value, i.e.,

$$\lim_{k \rightarrow \infty} x_i(k) = x_{av} := \frac{1}{n} \sum_{i=1}^n x_i(0). \quad (2.1)$$

We refer to this problem as the *average consensus problem*.

In gossip algorithms, each node has a clock which at random times chooses one of its neighbors to communicate its own state. The time a communication is attempted is called a transmission time  $k$  and assumed that each node has the same probability of being the node that initiated the communication. Such node, denoted by  $i$ , chooses a random out-neighbor  $j$  according to the probability distributions  $w_{i1}, w_{i2}, \dots, w_{in}$ ,  $\sum_{j=1}^n w_{ij} = 1$ ,  $\forall i$ . The set of all out-neighbors of  $i$  is denoted by  $\mathcal{N}_{out}(i)$ , with the number of elements in the set being given by  $n_{out}$ , and, equivalently, the set of all in-neighbors of  $i$  is denoted by  $\mathcal{N}_{in}(i)$ .

The communication topology is modeled by a directed graph  $G = (\mathcal{V}, E)$ , where  $\mathcal{V}$  represents the set of  $n$  agents, also called nodes, and  $E \subseteq \mathcal{V} \times \mathcal{V}$  is the set of communication links, also called edges. The node  $i$  can send a message to the node  $j$ , if  $(i, j) \in E$ . If there exists at least one  $i \in \mathcal{V}$  such that  $(i, i) \in E$ , we say that the graph has self-loops which can model, for example, packet drops since node  $i$  only has access to its own value at that transmission time. We associate to the

graph  $G$  a *weighted adjacency matrix*  $W$  with entries:

$$W_{ij} := \begin{cases} w_{ij}, & \text{if } (i, j) \in E \\ 0, & \text{otherwise} \end{cases}; w_{ij} \in [0, 1]$$

Our goal is to solve this problem using a *linear randomized gossip algorithm* defined by the iteration:

$$x(k+1) = U_k x(k), \quad (2.2)$$

where  $U_k$  is selected randomly from a set  $\{Q_{ij}, 1 \leq i \leq n, 1 \leq j \leq n\}$ . The matrices  $Q_{ij}$  implement the update on state variables  $x_i$  and  $x_j$  caused by a transmission from node  $i$  to node  $j$  and represent a set of *column stochastic* matrices (i.e.  $1^\top Q_{ij} = 1^\top$ ) to keep the average between iterations.

Since matrices  $U_k$  in (2.2) are randomly chosen, the state in (2.2) is a stochastic process and we need to specify how to interpret the convergence in (2.1).

**Definition 2.1** (Stochastic Convergence). *We say that the state of (2.2):*

(i) *converges almost surely to average consensus if*

$$\lim_{k \rightarrow \infty} x_i(k) = x_{av} := \frac{1}{n} \sum_{i=1}^n x_i(0), \quad \forall i \in \{1, \dots, n\}$$

*almost surely;*

(ii) *converges in expectation to average consensus if*

$$\lim_{k \rightarrow \infty} \mathbb{E}[x_i(k)] = x_{av}, \quad \forall i \in \{1, \dots, n\}.$$

(iii) *converges in mean square sense to average consensus if*

$$\lim_{k \rightarrow \infty} \mathbb{E}[(x_i(k) - x_{av})^2] \rightarrow 0, \quad \forall i \in \{1, \dots, n\}.$$

### 2.3.2 Proposed Solution

Our original proposed solution [ASS11] to the randomized gossip case is presented next where the key difference is stressed. We start by augmenting the original state  $x(k)$  with an auxiliary vector  $y(k) \in \mathbb{R}^n$ , and define

$$z = (x, y). \quad (2.3)$$

We consider a linear distributed algorithm taking the form

$$z(k+1) = U_k z(k), \quad (2.4)$$

where  $z(0) = (x(0), y(0))$ ,  $y(0) = 0$ . Intuitively, the purpose of  $y$  is to assure that at each iteration the total state average is kept constant, i.e., that

$$\frac{\sum_{i=1}^n x_i(k+1) + \sum_{i=1}^n y_i(k+1)}{2n} = \frac{\sum_{i=1}^n x_i(k) + \sum_{i=1}^n y_i(k)}{2n}. \quad (2.5)$$

## Chapter 2: Randomized Time-Dependent Algorithms

---

If we initialize  $y$  to zero and guarantee that  $y(k)$  goes to zero then average consensus is achieved. More specifically, the proposed algorithm can be described as follows.

At time  $k$ , a given node  $i$  sends a message containing  $x_i(k)$  and  $y_i(k)$  to one out-neighbor. The node  $i$  does not change its state, i.e.,

$$x_i(k+1) = x_i(k) \quad (2.6)$$

and resets the auxiliary state to zero

$$y_i(k+1) = 0. \quad (2.7)$$

A node  $j$  receiving this message, updates its state  $x_j(k)$  according to

$$x_j(k+1) = (1-\alpha)x_j(k) + \alpha x_i(k) + \beta y_j(k) + \gamma y_i(k) \quad (2.8)$$

and updates its variable  $y_j(k)$  according to

$$y_j(k+1) = \frac{y_i(k)}{n_{\text{out}}(i,k)} + y_j(k) + x_j(k) - x_j(k+1) \quad (2.9)$$

so that the total state average is kept constant, i.e., (2.5) holds. In the following sections, we present the details for each of the considered scenarios.

### Gossip algorithm $\mathcal{G}$

The matrices  $U_k$  are taken from the set  $\{Q_{ij}, 1 \leq i, j \leq n\}$ , where each  $Q_{ij}$  corresponds to a transmission from node  $i$  to an out-neighbor node  $j$ , and these matrices are described as follows. Let  $\Lambda_i := \text{diag}(\mathbf{e}_i)$  and  $\Omega_{ij} := I - (\Lambda_i + \Lambda_j)$ . Then

$$Q_{ij} = \begin{bmatrix} A_{ij} & B_{ij} \\ C_{ij} & D_{ij} \end{bmatrix} \quad (2.10)$$

where

$$\begin{aligned} A_{ij} &:= I - \alpha \Lambda_j + \alpha \mathbf{e}_j \mathbf{e}_i^\top \\ B_{ij} &:= \beta \Lambda_j + \gamma \mathbf{e}_j \mathbf{e}_i^\top \\ C_{ij} &:= \Lambda_j (I - A_{ij}) \\ D_{ij} &:= \Omega_{ij} + \Lambda_j (I + \mathbf{e}_j \mathbf{e}_i^\top - B_{ij}). \end{aligned} \quad (2.11)$$

The matrices defined in (2.10) also model the case where a node  $i$  picks itself when there is a clock tick (with probability  $w_{ii}$ ). The matrices  $U_k$  are by construction independent and identically distributed, and satisfy

$$\text{Prob}[U_k = Q_{ij}] = \frac{1}{n} w_{ij},$$

( $\frac{1}{n}$  is the probability that node  $i$  is the one whose clock ticks at  $k$  and  $w_{ij}$  the probability that  $i$  picks its out-neighbor node  $j$ ).

### Broadcast Algorithm $\mathcal{B}$

The matrices  $U_k$  are taken from the set  $\{R_i, 1 \leq i \leq n\}$ , where each  $R_i$  corresponds to a transmission from node  $i$  to every other node. Let  $\Lambda_i := \text{diag}(\mathbf{e}_i)$ ,  $\Omega_i = (I - \Lambda_i)$ . Then

$$R_i = \begin{bmatrix} A_i & B_i \\ C_i & D_i \end{bmatrix} \quad (2.12)$$

$$A_i = (1 - \alpha)I + \alpha \mathbf{1}_n \mathbf{e}_i^\top$$

$$B_i = \Omega_i(\beta I + \gamma \mathbf{1}_n \mathbf{e}_i^\top)$$

$$C_i = \Omega_i(I - A_i)$$

$$D_i = \Omega_i(I + \frac{\mathbf{1}_n \mathbf{e}_i^\top}{n-1} - B_i).$$

The matrices  $U_k$  are independent and identically distributed due to our assumption that nodes access the network with the same probability, i.e.

$$\text{Prob}[U_k = R_i] = \frac{1}{n}.$$

Hereafter, we denote by *gossip algorithm*  $\mathcal{G}$  the linear distributed algorithm modeled by (2.4) and (2.10), and denote by *broadcast algorithm*  $\mathcal{B}$  the linear distributed algorithm modeled by (2.4) and (2.12). Note that, by construction, for both gossip and broadcast algorithms the matrices  $\{U_k, k \geq 0\}$  are such that

$$\begin{bmatrix} \mathbf{1}_n^\top & \mathbf{1}_n^\top \end{bmatrix} U_k = \begin{bmatrix} \mathbf{1}_n^\top & \mathbf{1}_n^\top \end{bmatrix}, \quad (2.13)$$

which means that the total average is preserved at each iteration, i.e.,  $\mathbf{1}_{2n}^\top z(k+1) = \mathbf{1}_{2n}^\top z(k)$ , and

$$U_k \begin{bmatrix} \mathbf{1}_n \\ \mathbf{0}_n \end{bmatrix} = \begin{bmatrix} \mathbf{1}_n \\ \mathbf{0}_n \end{bmatrix} \quad (2.14)$$

which means that if consensus is achieved at iteration  $k$ , i.e., if  $x(k) = c\mathbf{1}_n$  and  $y(k) = \mathbf{0}_n$ , the state remains unchanged at iteration  $k+1$ , i.e.,  $x(k+1) = c\mathbf{1}_n$  and  $y(k+1) = \mathbf{0}_n$ .

## 2.4 Convergence Analysis

In this section, we provide results regarding the convergence for the two considered scenarios. We start by providing necessary and sufficient conditions to test the convergence of the algorithms for a particular network topology, which can be seen as a generalization of the bidirectional case for a unidirectional case with state augmentation.

The next theorem provides necessary and sufficient conditions for convergence of any of the algorithms with state augmentation.

**Theorem 2.1.** *Consider a linear distributed algorithm (2.4) where  $\{U_k, k \geq 0\}$  are characterized by (2.13), (2.14), and are randomly chosen from a set  $\mathcal{M} := \{B_i, 1 \leq i \leq n_p\}$ , according to*

$$\text{Prob}[U_k = B_i] = p_i, \sum_{i=1}^{n_p} p_i = 1.$$

## Chapter 2: Randomized Time-Dependent Algorithms

Then, the linear distributed algorithm converges in expectation to average consensus if and only if

$$r_\sigma\left(\sum_{i=1}^{n_p} p_i B_i - \frac{1}{n} \begin{bmatrix} 1_n \\ 0_n \end{bmatrix} \begin{bmatrix} 1_n^\top & 1_n^\top \end{bmatrix}\right) < 1 \quad (2.15)$$

and converges in mean square sense to average consensus if and only if

$$r_\sigma\left(\sum_{i=1}^{n_p} p_i B_i \otimes B_i - S\right) < 1, \quad (2.16)$$

where

$$S := \frac{1}{n^2} \left( \begin{bmatrix} 1_n \\ 0_n \end{bmatrix} \otimes \begin{bmatrix} 1_n \\ 0_n \end{bmatrix} \right) \left( \begin{bmatrix} 1_n^\top & 1_n^\top \end{bmatrix} \otimes \begin{bmatrix} 1_n^\top & 1_n^\top \end{bmatrix} \right).$$

□

*Proof.* We start by proving (2.15). Let  $R := \mathbb{E}[U_k] = \sum_{i=1}^{n_p} B_i$ . Since  $U_k$  are i.i.d matrices we have  $\mathbb{E}[z(k+1)] = R\mathbb{E}[z(k)]$ . By conditioning  $k$  times we have  $\mathbb{E}[z(k+1)] = R^k z(0)$ , from which we must have  $R^k \rightarrow \frac{1}{n} \begin{bmatrix} 1_n \\ 0_n \end{bmatrix} \begin{bmatrix} 1_n^\top & 1_n^\top \end{bmatrix}$  since we want  $z(k)$  to converge to consensus. By linearity of the expected value operator and since we defined  $B_i$  as in (2.13) and (2.14), we have an eigenvalue of 1 corresponding to the left eigenvector  $\begin{bmatrix} 1_n^\top & 1_n^\top \end{bmatrix}$  and to the right eigenvector  $\begin{bmatrix} 1_n \\ 0_n \end{bmatrix}$ . We need to have the remaining eigenvalues to have magnitude strictly less than 1 which gives (2.15).

To prove (2.16), we calculate the  $\mathbb{E}[z(k+1)z(k+1)^\top] = \mathbb{E}[U_k z(k)z(k)^\top U_k^\top]$ . Let us define  $Z(k) = z(k)z(k)^\top$ , then

$$\begin{aligned} Z(k+1)_{ij} &= ((U_k z(k))(U_k z(k))^\top)_{ij} \\ &= (U_k z(k))_i (U_k z(k))_j \end{aligned}$$

Defining  $\tilde{Z}(k) = \text{vec}(Z(k))$ , we have  $\tilde{Z}(k+1) = (U_k \otimes U_k) \tilde{Z}(k)$ . Let  $R_2 := \mathbb{E}[U_k \otimes U_k] = \sum_{i=1}^{n_p} B_i \otimes B_i$ . Since  $U_k$  are i.i.d,  $\mathbb{E}[\tilde{Z}(k+1)] = R_2 \mathbb{E}[\tilde{Z}(k)]$ , which by repeating the conditioning, we get  $\mathbb{E}[\tilde{Z}(k+1)] = R_2^k \tilde{Z}(0)$ . Thus,  $R_2^k \rightarrow S$  for the system to go to consensus. Take  $v = \begin{bmatrix} 1_n \\ 0_n \end{bmatrix}$  and  $w = \begin{bmatrix} 1_n^\top & 1_n^\top \end{bmatrix}$  and again due to linearity of the expected value operator,  $R_2$  has eigenvalue 1 for the right eigenvector  $v \otimes v$  and the left eigenvector  $w \otimes w$ . To have convergence all the remaining eigenvalues must have magnitude strictly less than 1 which gives (2.16). □

The previous theorem related the convergence with the spectral radius of a matrix for a given network topology and probabilities of communication. In the following theorem, we show that convergence holds for any strongly connected graph with symmetric communication probabilities.

**Theorem 2.2** (Convergence of  $\mathcal{G}$ ). *For any graph  $G$  which is strongly connected and admits a symmetric weighted adjacency matrix  $W$  the algorithm  $\mathcal{G}$  with parameters  $\alpha = \beta = \gamma = 1/2$  converges to consensus:*

- (i) almost surely;
- (ii) in expectation;
- (iii) in mean square sense.

□

*Proof.* We start by proving convergence (ii). Let

$$R := \mathbb{E}[U_k] = \sum_{i=1}^n \sum_{j \in \mathcal{N}_{\text{out}}(i)} w_{ij} Q_{ij}.$$

Since  $\mathbb{E}[z(k+1)] = R\mathbb{E}[z(k)]$  from the fact that  $U_k$  are independent, we have that

$$\mathbb{E}[z(k+1)] = \mathbb{E}\left[\begin{pmatrix} x(k+1) \\ y(k+1) \end{pmatrix}\right] = R^k z(0) = R^k \begin{pmatrix} x(0) \\ 0 \end{pmatrix}$$

and therefore it suffices to prove that

$$\lim_{k \rightarrow \infty} R^k = \frac{1}{n} \begin{bmatrix} \mathbf{1}_n \\ \mathbf{0}_n \end{bmatrix} \begin{bmatrix} \mathbf{1}_n^\top & \mathbf{1}_n^\top \end{bmatrix} \quad (2.17)$$

from which we conclude that  $\lim_{k \rightarrow \infty} \mathbb{E}[x(k+1)] = \mathbf{1}_n x_{\text{av}}$ ,  $x_{\text{av}} = \mathbf{1}_n^\top x(0)$ . From (2.10), (2.11) we notice that we can partition  $R$  into blocks  $R = \begin{bmatrix} R_1 & R_2 \\ R_3 & R_4 \end{bmatrix}$  where each block is a linear combination of the following three matrices

$$\begin{aligned} X &= \sum_{i=1}^n \sum_{j \in \mathcal{N}_{\text{out}}(i)} w_{ij} \Lambda_j, \quad Y = \sum_{i=1}^n \sum_{j \in \mathcal{N}_{\text{out}}(i)} w_{ij} \Lambda_i, \\ Z &= \sum_{i=1}^n \sum_{j \in \mathcal{N}_{\text{out}}(i)} w_{ij} \mathbf{e}_j \mathbf{e}_i^\top. \end{aligned}$$

It is easy to see that  $Z = W^\top = W$  (since we assume that matrix  $W$  is symmetric) and  $Y = I$ . Moreover,

$$X = \sum_{j=1}^n \sum_{i \in \mathcal{N}_{\text{in}}(j)} w_{ij} \Lambda_j = \sum_{j=1}^n \Lambda_j = I,$$

where we used the fact that  $\sum_{i \in \mathcal{N}_{\text{in}}(j)} w_{ij} = 1$ , i.e., the sum of weights for the in-neighbors of  $i$  equals to one, due to the key assumption that  $W : W_{ij} = w_{ij}$  is a doubly stochastic matrix. Therefore, each  $R_i$  is a linear combination of the matrices  $W$  and  $I$  and we can write

$$R = P_1 \otimes I_m + P_2 \otimes W,$$

where for  $\alpha = \beta = \gamma = \frac{1}{2}$ ,

$$P_1 = \begin{bmatrix} 1 - \frac{1}{2n} & \frac{1}{2n} \\ \frac{1}{2n} & 1 - \frac{3}{2n} \end{bmatrix}, \quad P_2 = \begin{bmatrix} \frac{1}{2n} & \frac{1}{2n} \\ -\frac{1}{2n} & \frac{1}{2n} \end{bmatrix}.$$

## Chapter 2: Randomized Time-Dependent Algorithms

---

We denote an eigenvalue of a matrix  $A$  by  $\lambda_i(A)$  and the set of eigenvalues by  $\{\lambda_i(A)\}$ . Let  $P_S(\delta) := P_1 + \delta P_2$ . Then one can obtain that

$$\lambda_i(P_S(\delta)) = 1 + \frac{\delta - 2 \pm \sqrt{2 - \delta^2}}{2n}, \quad i \in \{1, 2\}. \quad (2.18)$$

Let  $w_{P_i}$  be the two eigenvector of  $P_S(\delta)$ , and  $v_{P_j}$  denote the  $n$  eigenvectors of  $W$  (note that  $W$  is symmetric and therefore it has  $n$  eigenvectors). Then  $R$  has  $2n$  eigenvectors  $w_{P_i} \otimes v_{P_j}$ , since one can show that

$$R(w_{P_i} \otimes v_{P_j}) = \lambda_\ell(R) w_{P_i} \otimes v_{P_j}$$

where the set of eigenvalues of  $R$  is given by

$$\begin{aligned} \{\lambda_\ell(R), 1 \leq \ell \leq 2n\} &= \{\lambda_i(P_S(\eta_j)) : \eta_j \in \{\lambda_j(W)\}, \\ &1 \leq i \leq 2, 1 \leq j \leq n\} \end{aligned}$$

Since  $W$  is symmetric and doubly stochastic, and it is a weighted adjacency matrix of a strongly connected and aperiodic graph, the eigenvalues of  $W$  are real,  $W$  has a simple eigenvalue at 1, and all the remaining eigenvalues belong to the set  $(-1, 1)$ . Corresponding to the simple eigenvalue 1 of  $W$ ,  $R$  has two eigenvalues at  $\{\lambda_i(P_1 + P_2)\} = \{1, 1 - 1/n\}$ . Corresponding to the eigenvalues of  $W$  that belong to the set  $(-1, 1)$ , the eigenvalues of  $R$  are inside the unit circle. This can be shown by noticing that (2.18) is a strictly increasing function when  $-1 < \delta < 1$  for each  $i$  and, using this fact, it is easy to conclude that  $r_\sigma(P_1 + \delta P_2) < 1$  for  $-1 < \delta < 1$ . Thus  $R$  has a single eigenvalue at 1, all the remaining eigenvalues are inside the unit disk, and the vectors  $v_R := \begin{bmatrix} 1_n \\ 0_n \end{bmatrix}$  and  $w_R := \begin{bmatrix} 1_n^\top & 1_n^\top \end{bmatrix}$  are left and right eigenvalues of  $R$ , respectively, associated with this eigenvalue 1. This implies that

$$\lim_{k \rightarrow \infty} R^k = \frac{1}{w_R v_R} v_R w_R,$$

which is (2.17).

To prove (iii), let us introduce the shorter notation for the minimum and maximum as

$$x_{\min}(k) := \min_{\ell} x_{\ell}(k)$$

$$x_{\max}(k) := \max_{\ell} x_{\ell}(k),$$

and a Lyapunov function

$$V(x(k)) = x_{\max}(k) - x_{\min}(k).$$

Then, we have that  $\forall k \geq 0$

$$\begin{aligned} \|x(k) - x_{\text{av}} \mathbf{1}_n\|^2 &= \sum_{\ell=1}^n (x_{\ell}(k) - x_{\text{av}})^2 \\ &\leq (n-1)V(x(0)) \sum_{\ell=1}^n x_{\max}(k) - x_{\min}(k) \end{aligned} \quad (2.19)$$



where the inequality in (2.19) comes from the fact that, given the iteration defined in (2.8) and (2.9), any product of matrices  $Q_{ij}$  have a constant sum of entries equal to  $2n$  and any entry is not larger than 1. Combining these two facts, the maximum difference between two nodes is obtained when the row in the product of matrices  $Q_{ij}$  corresponding to  $\ell_{\max} := \arg \max_{\ell} x_{\ell}(k)$  is

$$\mathbf{e}_{\ell_{\max}} \begin{bmatrix} -(n-1) & \mathbf{1}_{2n-1}^T \end{bmatrix} + \begin{bmatrix} 1_n \\ 0_n \end{bmatrix} * \mathbf{e}_1^T$$

i.e., the  $x_{\max}(k) \leq (n-1)(x_{\max}(0) - x_{\min}(0)) \wedge x_{\min}(k) \geq x_{\min}(0)$  (and following the same reasoning,  $x_{\max}(k) \leq x_{\max}(0) \wedge x_{\min}(k) \geq x_{\min}(0) - (n-2)(x_{\max}(0) - x_{\min}(0))$  for the case of selecting the row in the product of matrices  $Q_{ij}$  corresponding to the  $x_{\min}(k)$ ). In both cases,  $V(x(k)) \leq (n-1)V(x(0))$ .

Using (2.19), it follows

$$\mathbb{E}[\|x(k) - x_{\text{av}} \mathbf{1}_n\|^2 | x(0)] \leq (n-1)V(x(0))\mathbb{E}[V(x(k)) | x(0)].$$

We shall prove that

$$\mathbb{E}[V(x(k)) | x(0)] \leq c\bar{\gamma}^k V(x(0)) \quad (2.20)$$

for a constant  $c$  from which stability in the mean square sense follows, because

$$\mathbb{E}[\|x(k) - x_{\text{av}} \mathbf{1}_n\|^2 | x(0)] \leq (n-1)c\bar{\gamma}^k V(x(0))^2$$

for some positive constant  $c$  and  $\bar{\gamma} < 1$ .

To prove (2.20), it is sufficient to show that

$$\mathbb{E}[V(x(k+\tau)) | x(k)] - \gamma V(x(k)) \leq 0 \quad (2.21)$$

for time interval of size  $\tau$ , constant  $\gamma < 1$ , which relates to  $\bar{\gamma}$  through  $\gamma^{\frac{k}{\tau}} = \bar{\gamma}^k$ , and where  $\mathbb{E}[\cdot]$  is the conditional expected value operator.

In order to upperbound the expected value in (2.21), we can define a finite sequence  $\theta$ , of size  $\tau$ , such that  $\theta_1 = U_{k+1}, \dots, \theta_{\tau} = U_{k+\tau}$ . Since by assumption the graph  $G$  is strongly connected and symmetric, there exists a path of nodes of at most  $n-1$  links that go from the maximum to the minimum-value nodes. Let us assume the longest path possible of  $n-1$  links and define the random variables  $\pi_1, \dots, \pi_n$  such that  $\pi_1(k) = x_{\max}(k)$  and  $\pi_n(k) = x_{\min}(k)$  with each  $\pi_i(k)$  being the  $i$ th node in the path from the maximum and minimum-value nodes at time  $k$ .

With the objective of writing  $x_{\min}(k+\tau)$  and  $x_{\max}(k+\tau)$  with terms that include both  $x_{\min}(k)$  and  $x_{\max}(k)$ , we consider a finite sequence, for the time instant  $k$ ,  $\theta^*$ . This sequence is constructed as follows  $\theta_1^* = Q_{\pi_1 \pi_2}, \theta_2^* = Q_{\pi_2 \pi_1}, \dots, \theta_{\tau-1}^* = Q_{\pi_{n-1} \pi_n}, \theta_{\tau}^* = Q_{\pi_n \pi_{n-1}}$ , where we omitted the dependence of  $\pi$  on  $k$  to improve readability. Therefore, each  $\theta_i^*$  is also a random variable as it depends on the path given by  $\pi$ . This sequence of updates, of size  $\tau = 2(n-1)$  occurs with non-zero probability

$$p_{\text{good}} = \frac{1}{n^{2(n-1)}} \prod_{\ell=1}^{n-1} ([W]_{\pi_{\ell}, \pi_{\ell+1}})^2,$$

## Chapter 2: Randomized Time-Dependent Algorithms

as all weights  $[W]_{\pi_\ell, \pi_{\ell+1}}$  are non-negative and  $[W]_{\pi_\ell, \pi_{\ell+1}} = [W]_{\pi_{\ell+1}, \pi_\ell}$ . Computing the product  $Q_{\pi_1 \pi_2} Q_{\pi_2 \pi_1} \cdots Q_{\pi_{n-1}, \pi_n} Q_{\pi_n, \pi_{n-1}} x(k)$ , the expected value of function  $V(\cdot)$  subject to the chosen sequence  $\theta^\star$  to occur from time  $k$  to  $k + \tau$  becomes

$$\begin{aligned} \mathbb{E}[V(x(k + \tau)) | x(k), \theta = \theta^\star] &= \frac{1}{2} x_{\pi_n}(k) + \frac{1}{2} x_{\pi_{n-1}}(k) \\ &\quad - \sum_{\ell=1}^{n-1} \left[ \frac{1}{2^\ell} x_{\pi_\ell}(k) \right] \\ &\quad - \frac{1}{2^{n-1}} x_{\pi_n}(k), \end{aligned} \quad (2.22)$$

where we draw attention for the fact that conditioning on  $x(k)$  means that the variable  $\pi$  becomes deterministic. We can upperbound (2.22) and get

$$\begin{aligned} \mathbb{E}[V(x(k + \tau)) | x(k), \theta = \theta^\star] &\leq x_{\pi_n}(k) - \\ &\quad \left[ \left( 1 - \frac{1}{2^{n-1}} \right) x_{\pi_1}(k) + \frac{1}{2^{n-1}} x_{\pi_n}(k) \right] \\ &\leq \left( 1 - \frac{1}{2^{n-1}} \right) (x_{\pi_n}(k) - x_{\pi_1}(k)) \\ &\leq \left( 1 - \frac{1}{2^{n-1}} \right) V(x(k)) \end{aligned}$$

where all the  $x_{\pi_\ell}(k)$  inside the summation in (2.22) were replaced by  $x_{\pi_1}(k)$ . Let us introduce the notation  $\Theta := \{\theta^\star\} \cup \Theta^n \cup \Theta^b \cup \Theta^c$  where  $\Theta$  is the set of all finite sequences of updates of size  $\tau$ ,  $\Theta^n$  is a subset of the sequences that do not increase the expected value,  $\Theta^b$  is the subset of sequences increasing the expected value in at most  $\vartheta$  for some constant  $\vartheta$ , and  $\Theta^c$  is the subset of sequences that decrease of at least  $\vartheta$ . Sets  $\{\theta^\star\}, \Theta^n, \Theta^b$  and  $\Theta^c$  are chosen to be mutually disjoint. Thus, the expected value in (2.21) can be written as

$$\begin{aligned} \mathbb{E}[V(x(k + \tau)) | x(k)] &= \sum_{\theta \in \Theta} p_\theta \mathbb{E}[V(x(k + \tau)) | x(k), \theta] \\ &= p_{\text{good}} \mathbb{E}[V(x(k + \tau)) | x(k), \theta = \theta^\star] \\ &\quad + \sum_{\theta^n \in \Theta^n} p_{\theta^n} \mathbb{E}[V(x(k + \tau)) | x(k), \theta = \theta^n] \\ &\quad + \sum_{\theta^b \in \Theta^b} p_{\theta^b} \mathbb{E}[V(x(k + \tau)) | x(k), \theta = \theta^b] \\ &\quad + \sum_{\theta^c \in \Theta^c} p_{\theta^c} \mathbb{E}[V(x(k + \tau)) | x(k), \theta = \theta^c] \end{aligned} \quad (2.23)$$

where  $p_\theta$  is the probability of occurring the finite sequence  $\theta$  out of all possible finite sequences of size  $\tau$ .

Let us define the random variables  $\pi_i^s(k)$  of length  $\rho$  as each representing a node in a sorted path of nodes. All sequences  $\theta^b$ , of size  $\tau = \rho + 1$ , are characterized by  $\theta_1^b = Q_{\pi_1^s \pi_2^s}, \dots, \theta_{\tau-1}^b = Q_{\pi_{\rho-1}^s \pi_\rho^s}, \theta_\tau^b = Q_{\kappa \pi_\rho^s}$ , for some node  $\kappa$  (once again to improve readability we omitted the dependence of  $\pi^s$  on  $k$ ).

We focus on showing that there is an equivalent sequence  $\theta^c$  with a greater or equal probability and decrease of the function  $V(\cdot)$  as that of  $\theta^b$ . Since matrix  $W$  is symmetric, the probability

$W_{ij} = W_{ji}$ , which means we can reverse paths and maintain the same probability. Also, the selection of matrices  $Q_{ij}$  is independent which makes probability of  $Q_{i_1 j_1} Q_{i_2 j_2}$  equal to  $Q_{i_2 j_2} Q_{i_1 j_1}$ . We must consider three cases:

- i)  $\kappa = \pi_\rho^s$  - i.e., failed transmission of the last node, which must be the minimum or the maximum;
- ii)  $\kappa = \pi_{\rho-1}^s$  - i.e., a sequence that ends in the minimum or the maximum;
- iii)  $\kappa \neq \pi_\rho^s$  - i.e., a communication from a node different than the minimum and maximum.

Let us construct a sequence  $\theta^c$ , of size  $\tau = \rho + 1$  for case i). Intuitively, the problem with i) is that the failed transmission forces the sum of the accumulated  $y$  variable with  $x$ . For i), we have  $\theta_1^c = Q_{\pi_\rho^s \pi_\rho^s}, \theta_2^c = Q_{\pi_1^s \pi_2^s}, \dots, \theta_\tau^c = Q_{\pi_{\rho-1}^s \pi_\rho^s}$ , where we changed the place of the failed transmission. In this case, we are in the same conditions then ii), which we address next, but for sequences of size  $\tau = \rho$ .

For case ii), if  $\pi_1^s \in \{x_{\min}(k), x_{\max}(k)\}$ , we can construct  $\theta_1^c = Q_{\pi_\rho^s \pi_{\rho-1}^s}, \dots, \theta_\tau^c = Q_{\pi_2^s \pi_1^s}$ , where we reversed the path. In doing so,  $p_{\theta^c} = p_{\theta^b}$  and the variation  $\vartheta$  for  $\theta^c$  is greater or equal than the variation for  $\theta^b$  since  $\pi_1^s - x_{\text{av}} \geq \pi_\rho^s - x_{\text{av}}$ . Intuitively, the bad case was due to nodes above the average contacting the minimum node which was closer to the average than the maximum, or vice-versa. If  $\pi_1^s \notin \{x_{\min}(k), x_{\max}(k)\}$ , we will have to consider all the sequences  $\pi_s$  of the same length entering  $\pi_\rho^s$ . Since  $W$  is symmetric, all the in-communications links sum to one and therefore, the probabilities of all sequences  $\pi^s$  for  $x_{\min}(k)$  have the same probability as the sequences  $\pi^s$  ending in  $x_{\max}(k)$  and the total variation is negative by the same reasoning.

Lastly, the construction for case iii) follows  $\theta_1^c = Q_{\pi_\rho^s \pi_{\rho-1}^s}, \dots, \theta_{\tau-1}^c = Q_{\pi_2^s \pi_1^s}, \theta_\tau^c = Q_{\pi_\rho^s \kappa}$ . The sequence  $\theta^c$  uses the same communicating pairs of nodes, so it happens with the same probability.

The main consequence is that

$$\sum_{\theta^b \in \Theta^b} p_{\theta^b} \leq \sum_{\theta^c \in \Theta^c} p_{\theta^c}.$$

Given that

$$\forall \theta^n \in \Theta^n : \mathbb{E}[V(x(k+\tau)) | x(k), \theta = \theta^n] \leq V(x(k)),$$

$$\forall \theta^b \in \Theta^b : \mathbb{E}[V(x(k+\tau)) | x(k), \theta = \theta^b] \leq V(x(k)) + \vartheta,$$

and

$$\forall \theta^c \in \Theta^c : \mathbb{E}[V(x(k+\tau)) | x(k), \theta = \theta^c] \leq V(x(k)) - \vartheta,$$

it is possible to overbound the terms in  $\theta^n$ ,  $\theta^b$  and  $\theta^c$  in (2.23) as  $(1 - p_{\text{good}})V(x(k))$  and get

$$\begin{aligned} \mathbb{E}[V(x(k+\tau)) | x(k)] &\leq p_{\text{good}} \left(1 - \frac{1}{2^{n-1}}\right) V(x(k)) \\ &\quad + (1 - p_{\text{good}}) V(x(k)). \end{aligned} \tag{2.24}$$

## Chapter 2: Randomized Time-Dependent Algorithms

---

By simplifying (2.24), we get

$$\mathbb{E}[V(x(k+\tau))|x(k)] \leq \left[1 - p_{\text{good}} \frac{1}{2^{n-1}}\right] V(x(k))$$

which satisfies (2.21) for  $\gamma = 1 - p_{\text{good}} \frac{1}{2^{n-1}}$ , getting convergence in mean square sense.

To prove i) notice that we verified ii) and iii), which means convergence for both the expected value and the expected value of the square occur with an exponential rate. Using the Borel-Cantelli first lemma [Bor09, Can17], the sequence converges almost surely.  $\square$

Let us recall the definition of *disagreement*  $\delta(x)$  [PS07] which is interesting for proving convergence for the broadcast algorithm.

**Definition 2.2** (disagreement). *For any vector  $x \in \mathbb{R}^n$ , let us define its disagreement  $\delta$  with respect to some norm  $\|\cdot\|$  as*

$$\delta(x) = \|x - \mathbf{1}_n x_{av}\|$$

*In particular, if using the  $\|\cdot\|_\infty$  and introducing the notation  $\bar{x} = \max_{i=1,\dots,n} x_i$  and  $\underline{x} = \min_{i=1,\dots,n} x_i$  we get*

$$\delta(x) = \frac{\bar{x} - \underline{x}}{2}$$

Definition 2.2 is particularly important to give properties about the evolution of the state in each iteration which we introduce in the following definition.

**Definition 2.3** (nonexpansive and pseudocontraction). *A matrix  $A \in \mathbb{R}^n$  is said to be nonexpansive if*

$$\|Ax - \mathbf{1}_n x_{av}\|_\infty \leq \|x - \mathbf{1}_n x_{av}\|_\infty$$

*which is equivalent to say that*

$$\delta(Ax) \leq \delta(x)$$

*and if the strict inequality holds then the matrix is a pseudocontraction.*

**Definition 2.4.** *A phase corresponds to an interval of time  $[\tilde{k}_\tau, \tilde{k}_{\tau+1})$  such that  $\exists k_i^\star \in [\tilde{k}_\tau, \tilde{k}_{\tau+1}), \forall i : 1 \leq i \leq n$ , node  $i$  transmits at time  $k_i^\star$ .*

The following lemma gives the nonexpansive behaviour using the time scale of phases for the algorithm  $\mathcal{B}$ .

**Lemma 2.1.** *For  $\lambda > 0$ , define  $S_\lambda = \{z \in \mathbb{R}^{2n} : \delta(z) < \lambda\}$ , where  $z$  is defined in (2.3) and satisfy the algorithm specified by (2.4) and equations (2.6), (2.7), (2.8) and (2.9). If  $z(0) \in S_\lambda$  then  $z(\tilde{k}_\tau) \in S_\lambda, \forall \tilde{k}_\tau > 0$  with probability 1. Equivalently*

$$\text{Prob} \left[ \sup_{0 \leq \tilde{k}_\tau < \infty} \delta(z(\tilde{k}_\tau)) \geq \lambda \right] = 0$$

*Proof.* From equations (2.6), (2.7), (2.8) and (2.9) and taking all the parameters to be  $1/2$ , we get that  $\forall i \in \mathcal{V}, \forall k > 0 : R_i^k = R_i$ . For the base case of a two-node network, and assuming without loss of generality that we label as node 1 the node that transmitted first, we will get  $z(\tilde{k}_1) = R_1^k R_2 z(0) = R_1 R_2 z(0) = \frac{1}{2} \begin{bmatrix} 1 & 2 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} z(0)$ . This implies  $\bar{z}(\tilde{k}_0) \geq \bar{z}(\tilde{k}_1)$  and conversely  $\underline{z}(\tilde{k}_0) \leq \underline{z}(\tilde{k}_1)$  which gives  $\delta(z(\tilde{k}_1)) \leq \delta(z(\tilde{k}_0))$ . The same reasoning is valid for subsequent  $\tilde{k}_\tau$ , thus meaning that for a 2-node network, we have  $\delta(z(\tilde{k}_{\tau+1})) \leq \delta(z(\tilde{k}_\tau))$ .

If we assume that  $\delta(z(\tilde{k}_{\tau+1})) \leq \delta(z(\tilde{k}_\tau))$  for any  $\tau$  and a network of  $n$  nodes, then let us prove the statement for a network of  $n+1$  nodes. Let us label node  $n+1$  as the last to transmit for the first time since  $\tilde{k}_\tau$ . By assumption, all the remaining nodes will have  $\delta(z_{-(n+1)}(\tilde{k}_{\tau+1})) \leq \delta(z_{-(n+1)}(\tilde{k}_\tau))$ , where the variable  $z_{-(n+1)}$  represents all the states except for the one of node  $n+1$ . Prior to time  $\tilde{k}_{\tau+1}$ , node  $n+1$  state is denoted by  $z_{n+1}(\tilde{k}_{\tau+1}^-)$  and in its  $x$  component it has  $x_{n+1}(\tilde{k}_{\tau+1}^-) \leq x_{n+1}(\tilde{k}_\tau)$  and in the  $y$  component a value  $\eta$  which is the difference changed in the  $x$  variable to keep the sum of the states constant. See that  $\eta < 0$  if  $x(0) < x_{av}$  and non-negative otherwise. When node  $n+1$  had  $x_{n+1}(0) < x_{av}$ , this implies that it will decrease the state variable of the remaining nodes on a proportion  $\frac{\eta}{n+1}$ . Therefore, the quantity  $\sum_{i=1}^n x_i - x_{av} + y_i$  decreases (as the sum of deviation above the average are greater than the deviations below the average when excluding the node  $n+1$  which directly implies that  $\delta(z(\tilde{k}_{\tau+1})) \leq \delta(z(\tilde{k}_\tau))$ ). Conversely, it also holds when the node  $n+1$  has  $x(0) \geq x_{av}$ . Then, by induction we have the property  $\delta(z(\tilde{k}_{\tau+1})) \leq \delta(z(\tilde{k}_\tau))$  for all  $n$  which proves the lemma.  $\square$

Based on Lemma 2.1 it is possible to state the following theorem regarding the convergence of  $\mathcal{B}$ .

**Theorem 2.3** (Convergence of  $\mathcal{B}$ ). *For any complete graph  $G$ , the algorithm  $\mathcal{B}$  with parameters  $\alpha = \beta = \gamma = 1/2$  converges to consensus:*

- (i) almost surely;
- (ii) in expectation;
- (iii) in mean square sense.

$\square$

*Proof.* We start by proving convergence in (ii) by showing that  $r_\sigma(R) \leq 1$ . We start by noticing that matrix  $R$  in this case can be rewritten as:

$$R = P_1 \otimes I_n + P_2 \otimes \frac{1_n 1_n^\top}{n}$$

where

$$P_1 = \begin{bmatrix} 1 - \alpha & \frac{-\gamma + (n-1)\beta}{n} \\ \alpha & \frac{n-2}{n-1} + \frac{\gamma - (n-1)\beta}{n} \end{bmatrix}, P_2 = \begin{bmatrix} \alpha & \gamma \\ -\alpha & \frac{1}{n-1} - \gamma \end{bmatrix}.$$

## Chapter 2: Randomized Time-Dependent Algorithms

Then,  $R$  has two simple eigenvalues in 1 and  $1/n$  and two eigenvalues with multiplicity  $n - 1$  corresponding to:

$$\lambda_i(R) = \frac{n - n^2 + 1 \pm \sqrt{n^4 - 4n^3 + 5n^2 - 2n + 1}}{2n(1 - n)}. \quad (2.25)$$

Using the derivatives of this expression for the eigenvalues in (2.25), we have  $\lambda_1 \in [\frac{1}{2}, 1]$  and  $\lambda_2 \in [0, \frac{5-\sqrt{13}}{12}]$ . Therefore,  $r_\sigma(R) \leq 1$ , which concludes the proof of convergence in expectation.

To establish (iii), let us select time instances as in the Definition 2.4 of phase and, by Lemma 2.1, the variable  $x$  is pseudocontracting meaning that  $x(\tilde{k}) \in S_\lambda, \lambda > 0$  and that the derivative is negative over phase intervals (also see [Mor04] and references therein). An equivalent formulation is that with probability one we have  $\forall \tilde{k} : V(x(\tilde{k})) \leq V(x(0))$ .

Given the definition of the function  $V(x(k)) := x_{\max}(k) - x_{\min}(k)$ , it holds that  $\forall k \geq 0$

$$\begin{aligned} \|x(k) - 1_n x_{\text{av}}\|^2 &= \sum_{\ell=1}^n (x_\ell(k) - x_{\text{av}})^2 \\ &\leq V(x(0)) \sum_{\ell=1}^n |x_\ell(k) - x_{\text{av}}| \\ &\leq V(x(0)) \sum_{\ell=1}^n x_{\max}(k) - x_{\min}(k) \end{aligned} \quad (2.26)$$

where the above inequalities come from the definition of maximum and minimum and  $\forall \ell \leq n, k \geq 0 : |x_\ell(k) - x_{\text{av}}| \leq x_{\max}(k) - x_{\min}(k)$ .

Using (2.26), it follows that

$$\mathbb{E}[\|x(k) - 1_n x_{\text{av}}\|^2 | x(0)] \leq n V(x(0)) \mathbb{E}[V(x(k)) | x(0)]. \quad (2.27)$$

However, given the result in (ii) we have that

$$\mathbb{E}[V(x(k+1)) | x(k)] \leq \zeta V(x(k))$$

for  $0 < \zeta < 1$ . Therefore, (2.27) becomes

$$\mathbb{E}[\|x(k) - 1_n x_{\text{av}}\|^2 | x(0)] \leq n \zeta^k V(x(0))^2$$

from which the conclusion follows.

The result in (i) is given by the exponential convergence in the mean square sense in (iii). In more detail, the Markon's Inequality states for a random variable  $X$  that

$$\mathbb{P}[X \geq a] \leq \frac{\mathbb{E}[X]}{a}.$$

If we define the error as  $e(k) = x(k) - 1_n x_{\text{av}}$ , we can compute

$$\begin{aligned} \lim_{k \rightarrow \infty} \mathbb{P} \left[ \frac{\|x(k) - 1_n x_{\text{av}}\|}{\|x(0) - 1_n x_{\text{av}}\|} \geq \epsilon \right] &= \lim_{k \rightarrow \infty} \mathbb{P} \left[ \frac{e(k)^\top e(k)}{e(0)^\top e(0)} \geq \epsilon^2 \right] \\ &\leq \lim_{k \rightarrow \infty} \epsilon^{-2} \frac{\mathbb{E}[e(k)^\top e(k)]}{e(0)^\top e(0)} \\ &= \lim_{k \rightarrow \infty} \epsilon^{-2} \zeta^k \\ &= 0. \end{aligned}$$

for the  $0 < \zeta < 1$  constant found for the convergence in mean square sense.

□

## 2.5 Convergence Rates

The interesting problem of finding the fastest distributed linear algorithm is addressed and the convergence rates are provided in discrete time. We show that the rates relate to the second largest eigenvalue of the linear combination of the transmission matrices. We start by providing a result available in the literature and showing how both algorithm  $\mathcal{G}$  and  $\mathcal{B}$  can be seen in that framework.

**Definition 2.5** ( $\epsilon$ -averaging time). *For any  $0 < \epsilon < 1$ , the  $\epsilon$ -averaging time, denoted by  $t_{avg}(\epsilon, p)$ , of a linear distributed algorithm (2.4), where  $\{U_k, k \geq 0\}$  are characterized by (2.13), (2.14), and randomly chosen from a set  $\mathcal{M} := \{B_i, 1 \leq i \leq n_p\}$ , where*

$$Prob[U_k = B_i] = p_i, \sum_{i=1}^{n_p} p_i = 1.$$

is defined as

$$\sup_{z(0)} \inf \left\{ t : Prob \left[ \frac{\|z(k) - z_{av} \mathbf{1}\|}{\|z(0)\|} \geq \epsilon \right] \leq \epsilon \right\}$$

where  $\|v\|$  denotes the  $l_2$  norm of the vector  $v$ .

Using the above definition, we provide the unidirectional version of the bounds found in [BGPS06].

**Theorem 2.4** (Convergence in discrete time). *The averaging time  $t_{avg}(\epsilon, p)$  (measured in terms of clock ticks) of the linear distributed algorithm, as defined in Definition 2.5 is bounded by:*

$$t_{avg}(\epsilon, p) \leq \frac{3 \log \epsilon^{-1}}{\log \lambda_2(R_2)^{-1}}$$

and

$$t_{avg}(\epsilon, p) \geq \frac{0.5 \log \epsilon^{-1}}{\log \lambda_2(R_2)^{-1}}$$

where

$$R_2 = \sum_{i=1}^{n_p} p_i B_i \otimes B_i$$

□

*Proof.* The proof follows from the fact that both algorithm  $\mathcal{G}$  and  $\mathcal{B}$  can be casted into the formulation of Definition 2.5 which is the same as the [Thm 3, [BGPS06]].

□

### 2.5.1 Distributed Optimization

In the previous section, we presented convergence results for the directed gossip algorithm. An important practical question is how we can optimize the rate of convergence given by the second largest eigenvalue. Such question is of interest because matrices  $Q_{ij}$  are nonsymmetric which renders the problem non convex.

**Theorem 2.5** (Distributed Optimization). *The directed gossip algorithm  $\mathcal{G}$  for a system of the form (2.4) with the linear iteration as in (2.10) can be optimized for communication probabilities in matrix  $W$  and for parameters  $\alpha$ ,  $\beta$  and  $\gamma$ .*

□

*Proof.* When optimizing for matrix  $W$  we are interested in solving the following optimization problem:

$$\begin{aligned} & \text{minimize} && \lambda_2(R) \\ & \text{subject to} && R = \sum_{i,j=1}^n \frac{1}{n} W_{ij} Q_{ij} \\ & && W_{ij} \geq 0, W_{ij} = 0 \text{ if } \{i, j\} \notin E \\ & && W \mathbf{1}_n = \mathbf{1}_n. \end{aligned}$$

However, notice that we used the fact that  $\lambda_i(P_S(\delta))$  is a monotonically increasing function with  $\lambda_2(W)$  to prove convergence, which allows us to rewrite the problem as:

$$\begin{aligned} & \text{minimize} && t \\ & \text{subject to} && W - \mathbf{1}_n \mathbf{1}_n^\top \leq t I_n \\ & && W_{ij} \geq 0, W_{ij} = 0 \text{ if } \{i, j\} \notin E \\ & && W \mathbf{1}_n = \mathbf{1}_n. \end{aligned}$$

Let us introduce for each directed link (the optimization can be carried out for nonsymmetric matrices  $W$ ) a new variable  $\eta_k$ ,  $1 \leq k \leq |E|$  and a correspondent flow matrix  $F_k = -(e_i - e_j)(e_i - e_j)^\top$  where the pair  $\{i, j\}$  is our  $k$ th link. The optimization can be written in the distributed form

$$\begin{aligned} & \text{minimize} && \lambda_2(I_n + \sum_{k=1}^{|E|} \eta_k F_k) \\ & \text{subject to} && L \mathbf{1}_n = \mathbf{1}_n \\ & && \eta_k \geq 0, 1 \leq k \leq |E|. \end{aligned}$$

Where the matrix  $L$  is just to short the notation and has  $L_{ij} = \eta_k$  for the corresponding  $k$  to the vertex  $\{i, j\}$  and zeros elsewhere. Using standard epigraph variable techniques and due to the



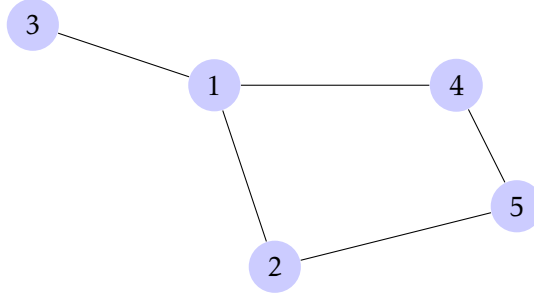


Figure 2.1: Communication graph with different out-neighbor degrees.

fact that  $\lambda(I_n + A) = 1 + \lambda(A)$

$$\begin{aligned} & \text{minimize} && t \\ & \text{subject to} && \sum_{k=1}^{|E|} \eta_k F_k - \mathbf{1}_n \mathbf{1}_n^\top \leq t I_n \\ & && L \mathbf{1}_n = \mathbf{1}_n \\ & && \eta_k \geq 0, \quad 1 \leq k \leq |E|. \end{aligned}$$

The formulation of the problem has separated optimization variables which can be performed in a distributed fashion using techniques such as Alternating Direction Method of Multipliers (ADMM) [BPC<sup>+</sup>11] or other techniques (see [BDX03] and [Lew96] and references therein).

Regarding parameters  $\alpha$ ,  $\beta$  and  $\gamma$ , the optimization is non-convex and can be carried using brute force both for the eigenvalues of the expectation and mean square matrices using the sufficient and necessary conditions presented in Theorem 2.1.  $\square$

### 2.5.2 Comparison between unidirectional and bidirectional case

In the previous section, we showed how to optimize the probabilities and parameters of the gossip algorithm and how to distribute that computation over the nodes of the network. It is interesting to compare how the convergence rate is affected when going from bidirectional to unidirectional gossip randomized algorithms. In the sequel, we present results about 3 communication graphs with different out-neighbor degree to give a general overview.

In selecting different cases to illustrate how the second largest eigenvalue of the matrix of expected value and second moment varies with parameters choice, we took into consideration what should be the best and worst case scenarios and an *average case*, where we are by no means stating that the chosen case is the average case, since our aim is to give an example with different out-neighbor degrees. Figure 2.1 presents the graph which we called as the average case example. The best case scenario is when connectivity is at its maximum (i.e. each node can communicate to every other node) and the worst case is when node  $i$  is connected to nodes  $i - 1$  and  $i + 1$  except for node 1 and  $n$  which connect to only one neighbor. For all the examples, we take the number of nodes  $n = 5$ .

Provided that the nodes optimize the matrix of probabilities  $W$  we get the following results:

$$W_{\text{best}} = \frac{1}{n-1}(\mathbf{1}_n \mathbf{1}_n^\top - I_n),$$

$$W_{\text{worst}} = \frac{1}{2}(\text{tri}(n) - I_n + \mathbf{e}_1 \mathbf{e}_1^\top + \mathbf{e}_n \mathbf{e}_n^\top),$$

$$W_{\text{average}} = \begin{bmatrix} 0 & 0.2835 & 0.433 & 0.2835 & 0 \\ 0.2835 & 0.2165 & 0 & 0 & 0.5 \\ 0.4330 & 0 & 0.567 & 0 & 0 \\ 0.2835 & 0 & 0 & 0.2165 & 0.5 \\ 0 & 0.5 & 0 & 0.5 & 0.0000 \end{bmatrix}.$$

where  $\text{tri}(n)$  is a tridiagonal matrix of size  $n$  with the elements in the three main diagonals all equal to 1.

Using the computed matrix  $W$  for the probabilities, we calculate the second largest eigenvalue for both Expectation and Second Moment which are presented in Table 2.1 for the 3 considered cases. Those values were obtained by searching in a brute force fashion for  $\alpha, \beta, \gamma \in [0, 1]$  which minimized  $\lambda_2$ . Regard, however that the minimum for the Expectation and Second Moment were not obtained jointly since one may wish to optimize for one or the other.

	Expectation		Second Moment	
Case	$\text{b-}\lambda_2$	$\text{u-}\lambda_2$	$\text{b-}\lambda_2$	$\text{u-}\lambda_2$
<i>Best</i>	0.75	0.76	0.75	0.9153
<i>Average</i>	0.9401	0.9625	0.9401	0.97
<i>Worst</i>	0.9618	0.9805	0.9618	0.983

Table 2.1: Second largest eigenvalue for the bidirectional ( $\text{b-}\lambda_2$ ) and the presented unidirectional ( $\text{u-}\lambda_2$ ) algorithms for the 3 studied cases and for the Expectation and Second Moment.

In order to give a better perspective about the values in Table 2.1, let us compute the upper and lower bound of clock ticks so that the system is in a neighborhood  $\epsilon$  of the solution  $x_{\text{av}}$ . Such bounds were provided in [BGPS06], although see references therein for additional information. The convergence rate in continuous time is provided in Theorem 9 and Corollary 10 in [ASS11]. It is important to notice that, in reality, a bidirectional algorithm is using two communication steps in each transmission so the values presented in Table 2.2 for the bidirectional case should be seen in a unit of measure which is double from the unidirectional case.

## 2.6 Conclusions

In this chapter, the problem of studying the convergence of the state of an average consensus algorithm with unidirectional communications is tackled. The motivation behind constructing an asynchronous and unidirectional algorithm was to better map the characteristics of wireless networks. In doing so, the algorithm can progress to the average of the initial values even in a

Case	Lower bound		Upper bound	
	b_ticks	u_ticks	b_ticks	u_ticks
<i>Best</i>	8	26.02	48.02	156.1
<i>Average</i>	37.28	75.6	223.66	453.57
<i>Worst</i>	59.12	134.29	354.71	805.75

Table 2.2: Upper and lower bounds for the mean square on the number of ticks for the algorithms to reach in a neighborhood of the solution of  $\epsilon = 10^{-2}$  for the bidirectional case (b\_ticks) and the presented unidirectional (u\_ticks) algorithms.

realistic scenario with a high packet drop rate as it can use the received information instead of having to wait for a successful two-way communication.

We firstly provide results to test the convergence for a specific instance of the connectivity graph for a generic algorithm obeying the definitions for the interactions. These relate to determining if the spectral radius of the matrix defining the expected value and the second moment is the unit circle. It is then shown that convergence holds for any connectivity graph that is symmetric.

Selecting the fastest converging algorithm for the average consensus problem is also presented in this chapter. By noticing that the spectral radius depends monotonically with the second largest eigenvalue of the expected value matrix, allowed us to first rewrite that optimization as a Semi-definite program and then optimize in a brute force fashion for the parameters of the algorithm. The convergence rate is compared to both the unidirectional and bidirectional case.



# 3

## RANDOMIZED STATE-DEPENDENT ALGORITHMS

### 3.1 Introduction

Time-dependent algorithms present limitations in modeling more complex decisions and communication interactions between the agents in a network. By having a network setup that is independent from the state of the system brings the possibility for using stochastic analysis tools requiring events to be independent. The powerful theory to provide convergence results is a major advantage at the expenses of having more complex behaviors that can result in significant improvements in the convergence rate.

On the other hand, in cases where the state is related to the position of the agent, the wireless network links depend on the distance between nodes among other factors which are inherently state-dependent. In simplifying a model by discarding these type of actions, a protocol design might be losing important features such as finite-time convergence properties with the same type of tolerance to faults.

This chapter focus on the particular case of social networks as a building block for other interesting cases where the state of the system contains the position of the agents. By analyzing the impact of the evolution of the network, a protocol designer can drive the system to a final conclusion with agents initial states having different contributions. Tolerance to network faults is complemented by adding stochastic interactions between pair of nodes and a distinct pallet of tools is given to provide convergence results.

### 3.2 Main Contributions and Organization

The chapter is organized as follows. Initial attention is given to the case of a social network where the network dynamics are state-dependent. We then progress to show that, for this case, it is still possible to design stochastic versions of the deterministic algorithms that inherit its performance. By appropriately selecting the parameters of the network, the designer can drive

the system to different configurations that depend on the initial conditions of nodes closest to the minimum and maximum or the median.

The analysis of the social network within a political party or an association, where agents are rational when evaluating each argument of other nodes, and where influence occurs among agents whose opinion is closer, makes possible the interesting contributions (presented in the papers [SRHS15b] and [SRHSed]):

- The social network is modeled as an iterative distributed algorithm, where the network is state-dependent with a fixed parameter of maximum number of connections, both in the deterministic and stochastic senses;
- Considering only nodes with distinct opinion is shown to require half the number of neighbors to obtain finite-time convergence;
- In the case of asymptotic convergence, the social opinion is shown to depend on the left eigenvector of a matrix, both in the deterministic and stochastic senses and, under certain assumptions, it is shown that the opinion achieves the average of the initial values;
- Finally, two strategies are investigated — one where nodes with extreme opinions contact each other, and another where agents require a fixed number of neighbors — and proved to converge in finite time, even when the number of neighbors is restricted to two other nodes, and the social opinion is shown to depend more on the minimum and maximum opinion nodes, for the first strategy, and on the nodes closest to the median, for the second one.

### 3.3 Social Networks

#### 3.3.1 Motivation

The study of social networks relates to understanding the mechanisms used in a group of agents to decide about a given issue. In particular, focus is given to determine the key agents that contribute the most to driving the general opinion of the network to a certain desired final state. In another direction, importance is given to identifying the general properties of the social network that ensure convergence of opinion given a model with iterative dynamics, representing the interaction between agents along time. This chapter addresses the problem of showing convergence for a state-dependent social network and the impact that small changes on the way the nodes interact have in the convergence time. Such observations are interesting in practical terms, in the sense that marketing campaigns can benefit from having proper information dissemination significantly reducing the convergence time. In [SRHS15b], preliminary results about convergence are given for the deterministic case. In this chapter, those results are extended to the stochastic case by showing the converse results in terms of expected value, when the nodes communicate in a random fashion.

In this chapter, we deal with social networks in a political party or a group where people contact a subset of the group with similar opinions on a subject. It is assumed that these opinions describe objective arguments and that people are rational and will take into consideration all received opinions regardless of the person who sent them. A similar terminology of *rational* innovations is used in [Kra97] where the opinion of an agent towards an innovation is *rational* if it depends only on the quality of the innovation, as opposed to *controversial* innovations. The same problem can be found in different scenarios. As an example, consider a group of people discussing the location to rendezvous, equipped with communication devices that have a variable power to transmit. To avoid the cost of transmitting to other people that are further away from their location, the subjects can only contact a small nearby subset. In addition, deciding on the final location depends only on the position of each agent and not on who is at which place.

An example within a different framework is a social network such as Facebook. One can consider an application to make a pool where players are paired with other members to discuss a given topic. The pairing serves the purpose of preventing a person from disregarding an opinion just because it is completely the opposite of its own. If the pool is about a product being marketed, it is of interest to study what is the effect of different pairings on the final opinion of the network. In a company environment, the same type of problem can be observed if the manager wants to keep all the workers satisfied, in which case it is interesting to understand how the pairing between different workers with different levels of work satisfaction would impact the final level of happiness.

Another motivation in the field of control for mobile networks is that one might want to replicate a social behavior in a distributed system by enforcing the same rules for neighbor selection. A group of mobile robots agreeing on the location to rendezvous, equipped with communication devices of variable transmitting power, would have crucial features such as: saving resources, as nodes limit the number of interconnections; having fast convergence when compared to solutions with asymptotic convergence; working both synchronously and asynchronously; and the generated network topology is regular and robust to link failures. These illustrative scenarios motivate the following problem.

### 3.3.2 Related Work

In [JMFB13], the authors study a classical model of influence networks and opinion formation processes found in sociology, which considers the evolution of power of each agent based on previous opinion formation process outcomes. The focus is on finding out how the weights assigned to each agent evolve if they are constructed using the previous relevance of a specific node and corresponding previous weight. The analysis focuses on the convergence properties of the model of Friedkin-Degroot [Deg74], [Fri11], which models social interactions by means of a linear system, where each agent updates its opinion as a weighted average of their previous

opinion and that of their neighbors.

The main observation is that people in social networks such as a political party, a sports association, or any other organization, tend to contact agents with similar opinions. The work of [HK02] and [WDAN02] points to the same conclusion. In particular, [HK02] studies various models of interaction to analyze when nodes converge to the same opinion, tend to polarize, or fragment into various opinion clusters that do not communicate. In [WDAN02], a model is investigated where, as in a gossip fashion, randomly selected pairs of nodes interact as long as their opinions are close. Nodes average their beliefs and the cases where they converge to a single or to multiple opinion clusters are investigated. Both works share a common view that the connectivity graphs depend on the state. The assumption here is that each node is only allowed to have a fixed number of influence connections, which is motivated by the fact that people have a limited number of acquaintances and, in their decision process, consider a small number of agents to form their opinion.

An interesting topic in social networks is to prove convergence of the opinion in the presence of leaders, who try to drive the remaining agents to a certain final value. In [KKPD13], this problem is addressed assuming that the network is state-dependent in the sense that communication occurs between nodes where the difference of their state is below a certain threshold. The problem is recast as a networked fractional-order system whose stability is studied. Using the fact that the initial graph has a directed spanning tree, the authors of [KKPD13] provide a potential function so as to get the system to consensus, while maintaining network connectivity. The main difference between [KKPD13] and the work presented in this thesis is the study on how the definition of the state-dependent rule can influence the speed of convergence, by using an alternative approach, based on recent advances in consensus methods available in the literature. In addition, we also show how the definition of the state-dependent rule can influence the final opinion in the presence of leaders. A more recent work [Fri15] studies the community cleavage problem as the result of stubborn leaders. A more comprehensive discussion of this topic can be found in [PT17].

In [FLZJ13], the problem of selecting leaders is considered by determining which nodes contribute the most to both the steady and the transient states (see [Fri91] for a seminal work on the application of centrality measures to determine influence in social networks). Different metrics for social influence of the nodes are presented, allowing the construction of a non-convex optimization problem for the optimal leader selection problem. Convex relaxation techniques are employed and a distributed solution is found. The authors also consider how to add social interactions to maximize the impact on the social influence of the set of leaders. Note that [FLZJ13] is related to this work in the sense that not only the final opinion value is important, but also how fast the agents reach that opinion. However, the dynamics assumed in [FLZJ13] are time-invariant, whereas a more generic framework is considered herein, which is able to account for time-domain changes in the network structure.



The study of having antagonistic links in the network can be found in the literature and the tools used for the analysis are common to the work presented herein. Examples like [AL15] and [Alt13], where a network is considered with some agents influencing negatively some of their neighbors, discusses the predictable formation of opinion. The work presented in this chapter does not include this possibility, but rather focuses on the communication graph dynamics. The topic of antagonistic links is left as a future research path.

Randomized algorithms for information aggregation have attracted attention due to its decentralization and accurate modeling of people interactions. In particular, [TFNM13] generalizes the concept for a set of agents with a state that reflects many opinions on different topics. This can be seen as a generalization of the randomized gossip algorithm proposed in [BGPS06] and the top- $k$  selective gossip [UR12], which encompasses other interesting particular cases such as for political voting, as mentioned in [TFNM13]. The work presented here differs from [TFNM13] in the sense that the evolution of the network is deterministic, motivated by having an environment with a set of rules and where people are rewarded for their cooperation. Another example of the study of a stochastic social network can be found in [FRTI13], [RFTI15], where a model of affine dynamics is studied under stochastic interactions. The present work differs from these models by assuming a different update rule and focusing on having network dynamics that mimic social interactions.

The topic of convergence of social networks is closely related to that of distributed linear iterative consensus (see, e.g., [OSM04], [BCM09], [HSJ14], [CHT14], and [DGH13]). The dynamic system generated has similarities and most tools used in the convergence proofs are common to both fields [TN14]. Research interest has risen in the study of stochastic packet drops and link failures [PBEA10], the existence of delays [HC14], [FZ09], quantized data transmissions [CBZ10], state-dependent noise [LWZ14], and time-varying communication connectivity [OSM04], [CI14]. In [CI11], the authors assume randomized directional communication in a consensus system. Some of these concepts have counterparts in the analysis of social networks. The work of [SJ13] addresses the problem of consensus with state-dependent dynamics and the tools to obtain the proofs are similar to those adopted in this chapter.

When addressing convergence, a meaningful characterization will describe the rate at which the process reaches the final value. For the average consensus problem, [FZ08] analyzes the examples of complete and Cayley graphs with tools based on computing the expected value of the difference between the state and the average. These results follow a similar reasoning to what is presented in this chapter for the stochastic social network (for the deterministic case, we follow another line-of-proof, as the objective is to get a finite number of steps instead of a asymptotic convergence rate). The main difference between the approach provided in this chapter and that of [FZ08] is the focus on a different Lyapunov function, since the final consensus value is not known *a priori*.

### 3.3.3 Problem Statement

We consider a social network where a set of  $n$  agents, also called nodes, interact and influence each other about a personal belief or opinion regarding some subject or discussion topic. The belief of agent  $i$  is denoted by a scalar  $x_i(k)$ ,  $1 \leq i \leq n$ , where we consider the time as a discrete variable  $k$ , which is incremented whenever agents communicate among themselves and their beliefs are updated.

The objective is to determine the final belief of the social network,  $x_\infty$ , defined as

$$x_\infty := \lim_{k \rightarrow \infty} x(k)$$

provided that the above limit exists.

The network of interconnections representing the influence that each agent has over another agent is modeled by a time-varying directed graph  $G(k) = (\mathcal{V}, E(k))$ , where  $\mathcal{V}$  represents the set of  $n$  agents, also denoted by nodes, and  $E(k) \subseteq \mathcal{V} \times \mathcal{V}$  is the set of influence links that change over time. Node  $i$  influences the opinion of node  $j$ , at time  $k$ , if  $(i, j) \in E(k)$ .  $\mathcal{N}_i(k)$  represents the set of neighbors of agent  $i$ , i.e.,  $\mathcal{N}_i(k) = \{j : (j, i) \in E(k)\}$ .

The set of edges  $E(k)$  evolves according to a “nearest” policy which is motivated by agents searching for a diverse set of opinions. In real-life, when people want to make a decision, they search for positive and negative feedback within other nodes with opinion similar to the node state [HK02], [WDAN02], with a constraint on the amount of feedbacks they can read or consult. A social network such as Facebook, a gaming platform, or another application, connect people based on their skills and opinions with both people with higher and lower rank values.

We avoid the standard approach to model the agent update rule of its state as a consensus-like problem (see, for instance, [Deg74], [Fri11] for the deterministic consensus-like dynamics and [FRTI13], [RFTI15] for the stochastic counterpart). Instead, we envisage a social network where the opinion translates a set of arguments. In [SST93], a comprehensive discussion on how a decision opinion is based on the positive arguments compensating the negative ones, is presented, which motivates to consider the average between the worst and the best sets of arguments. Agents are objective, i.e., *rational* in the nomenclature of [Kra97], meaning that, at time  $k$ , all nodes would reach the same conclusion if they had access to all the remaining opinions. Notice that the way nodes evaluate the arguments can change over time. These observations translate into the following dynamics for agent  $i$

$$x_i(k+1) = \alpha_k \min_{j \in \mathcal{N}_i(k)} x_j(k) + (1 - \alpha_k) \max_{j \in \mathcal{N}_i(k)} x_j(k) \quad (3.1)$$

where parameter  $\alpha_k \in [0, 1]$  models how the agents balance their conclusions with respect to the extreme (minimum and maximum) opinions of their neighbors. Note that the minimum and maximum are well-defined as the set  $\mathcal{N}_i(k) \neq \emptyset, \forall k$ , since at least the node itself is in the neighbor set. In the deterministic definition of the social network, all agents update synchronously their

opinion whereas we will present the details of the random selection for the stochastic case later in this chapter.

Parameter  $\alpha_k$  represents the level of optimism/pessimism of the agents, which is assumed to take the same value for all the members of the network. Associating a positive stance to high values of the belief, then  $\alpha_k = 0$  would correspond to optimistic agents that only take into account beliefs more positive than their own, whereas  $\alpha_k = 1$  would correspond to pessimistic agents. When considering a single value  $\alpha_k$  for all the nodes, focus is being given to a specific type of decision-making. However, it is also interesting to study the case where each node might have a different value. Apart from the asymptotic convergence in the deterministic and stochastic cases, the proofs of the theorems would no longer be valid. In future work, it is of relevance to consider different values for  $\alpha_k$ . In particular, extending the results in this chapter would characterize under what circumstances there is still finite-time convergence.

The problem described in this section can be summarized as that of determining whether the opinions of the agents will converge. Moreover, for practical applications, it is often useful to know if the desired convergence to a consensus is met in finite-time, i.e.,

$$\exists k_f : \forall k \geq k_f, i, j \in \mathcal{V}, |x_i(k) - x_j(k)| = 0.$$

When the previous condition is met, one would like to determine the smallest  $k_f$  as a condition on the number of nodes  $n$  and neighbors  $\eta$ . On the other hand, *asymptotic convergence* is obtained if

$$\forall i, j \in \mathcal{V}, \lim_{k \rightarrow \infty} |x_i(k) - x_j(k)| = 0.$$

We are also interested in comparing different definitions for the graph dynamics to determine key features influencing the rate of convergence and final opinion shared by the nodes. We start by introducing the deterministic version of the network dynamics and then progress to analyze the stochastic setting which reflects more accurately other real-life examples.

### 3.4 Neighbor Selection Rules

In order to get a simple definition, we introduce the notation for permutation  $\{(i) : i \in \mathcal{I}\}$  of the indices in the index set  $\mathcal{I}$  such that  $x_{(i)}(k) \leq x_{(i+1)}(k)$  and  $x_{(i)}(k) = x_{(i+1)}(k) \implies (i) < (i+1)$  (i.e., the permutation  $(i)$  is such that all the opinions become sorted and when two opinions are equal the sorting is resolved by the indices of the nodes). Based on this permutation of an index set, we have the following definition.

**Definition 3.1** (order of). *Take a node  $i$  and a set  $\mathcal{S}$  of indices for which we have a permutation  $(j)$  as before. We define that  $j$  is the order of  $i$  in the set  $\mathcal{S}$  if  $(j) = i$ .*

We can now present four definitions for neighbor selection that aim at capturing different behaviors. With a slight abuse of notation, we will use  $N_i(k)$  and redefine it. The reader can

### Chapter 3: Randomized State-Dependent Algorithms

---

recognize  $N_i(k)$  as the set of in-neighbors of  $i$  and, in each result, the appropriate definition is referred. The following definition uses the set  $\mathcal{V}_i(k) := \{\ell : x_\ell(k) \neq x_i(k)\} \cup \{i\}$ .

**Definition 3.2** (base network). *For each node  $i \in \mathcal{V}$  of order  $j$  in the set  $\mathcal{V}_i(k)$ , we define the set of at most  $\eta$  neighbors with opinion smaller than that of  $i$  as  $N_i^-(k)$ , i.e.,*

$$N_i^-(k) = \begin{cases} \{(j-\eta), (j-\eta+1), \dots, (j)\}, & \text{if } j-\eta \geq 1 \\ \{(1), (2), \dots, (j)\}, & \text{otherwise.} \end{cases}$$

and the set of at most  $\eta$  neighbors with higher opinion  $N_i^+(k)$  defined as

$$N_i^+(k) = \begin{cases} \{(j), (j+1), \dots, (j+\eta)\}, & \text{if } j+\eta \leq n \\ \{(j), (j+1), \dots, (n)\}, & \text{otherwise.} \end{cases}$$

and the set of all neighbors as  $N_i(k) := N_i^-(k) \cup N_i^+(k)$ , where  $\eta \in \mathbb{Z}^+$ .

□

Notice that  $0 < |N_i(k)| \leq 2\eta + 1$ , so no assumption is made on the degree of the nodes in  $G(k)$ .

The node selection policy outlined in the previous definition may lead to slow convergence because nodes near the minimum or maximum values of the belief have fewer links, as either the set  $N_i^-(k)$  or  $N_i^+(k)$  has cardinality smaller than  $\eta$ . While in real social networks, people with extreme opinions may indeed interact with less neighbors precisely because of their extreme views, it is still interesting to study how deviations from the policy outlined in the previous section may lead to faster convergence.

In real-life, the next policy is observed when people disregard the opinions of some of their acquaintances because they know that two individuals share the same positive or negative points towards the subject being discussed. In a different direction, one can resort to this definition in distributed systems or virtual social networks (such as Facebook) to reduce resource allocation by removing connections to neighbors that share the same opinion. Before introducing the proposed network dynamics, it is useful to consider the set of neighbors with distinct values. In particular, we denote by  $\mathcal{D}_i(k)$  the set of distinct possible neighbors of node  $i$  at time  $k$ , i.e., obtained by going through all the elements of  $\mathcal{V}_i(k)$  and adding them to  $\mathcal{D}_i(k)$  if there does not exist an element in  $\mathcal{D}_i(k)$  already with equal state. In doing so, for all the nodes with duplicate state, there exists at least one in  $\mathcal{D}_i(k)$ .

**Definition 3.3** (distinct value). *For each node  $i \in \mathcal{V}$  of order  $j$  in the set  $\mathcal{D}_i(k)$ , we define the set of at most  $\eta$  neighbors with opinion smaller than that of node  $i$  as  $N_i^-(k)$ , i.e.,*

$$N_i^-(k) = \begin{cases} \{(j-\eta), (j-\eta+1), \dots, (j)\}, & \text{if } j-\eta \geq 1 \\ \{(1), (2), \dots, (j)\}, & \text{otherwise.} \end{cases}$$

and

$$N_i^+(k) = \begin{cases} \{(j), (j+1), \dots, (j+\eta)\}, & \text{if } j+\eta \leq n \\ \{(j), (j+1), \dots, (n)\}, & \text{otherwise.} \end{cases}$$

and define the set of all neighbors  $N_i(k) := N_i^-(k) \cup N_i^+(k)$ .

□

By only counting distinct neighbors (i.e., nodes with distinct beliefs) we focus our attention on policies where nodes seek to be informed by a diversified set of opinions in their decision processes. Such a network has the structure depicted in Figure 3.2b.

The previous definition did not take into account the behavior of some people that want to assure an informed decision and therefore get exactly  $2\eta$  neighbors. One of their possibilities is to look for other closer nodes which motivates a second network structure (or policy), referred to as *nearest distinct neighbors*, being defined as follows:

**Definition 3.4** (distinct neighbors). *For each node  $i \in \mathcal{V}$  of order  $j$  in the set  $\mathcal{D}_i(k)$ , we define the set of at most  $\eta$  neighbors with opinion smaller than that of node  $i$  as  $N_i^-(k)$ , i.e.,*

$$N_i^-(k) = \begin{cases} \{(j-\eta), \dots, (j)\}, & \text{if } j-\eta \geq 1 \wedge j+\eta \leq n \\ \{(1), (2), \dots, (j)\}, & \text{if } j-\eta < 1 \wedge j+\eta \leq n \\ \{(\max\{1, n-2\eta\}), \dots, (j)\}, & \text{otherwise.} \end{cases}$$

and

$$N_{(i)}^+(k) = \begin{cases} \{(j), (j+1), \dots, (j+\eta)\}, & \text{if } j+\eta \leq n \wedge j-\eta \geq 1 \\ \{(j), (j+1), \dots, (n)\}, & \text{if } j+\eta > n \wedge j-\eta \geq 1 \\ \{(j), \dots, (\min\{n, 2\eta+1\})\}, & \text{otherwise.} \end{cases}$$

and define the set of all neighbors  $N_i(k) := N_i^-(k) \cup N_i^+(k)$ .

□

In this definition, nodes correct their lower degrees by contacting with other nearest neighbors (see Figure 3.1c). The next definition is somehow counterintuitive as nodes contact with others with opposite opinions to correct their lower degree. Even though the behavior of this strategy is completely different from the previous one, it establishes that convergence rate is governed by the ability to form clusters, i.e., a group of nodes sharing a common opinion.

**Definition 3.5** (circular value). *For each node  $i \in \mathcal{V}$  of order  $j$  in the set  $\mathcal{D}_i(k)$ , we define the set of at most  $\eta$  neighbors considered as  $N_i^-(k)$  as*

$$N_i^-(k) = \begin{cases} \{(j-\eta), \dots, (j)\}, & \text{if } j-\eta \geq 1 \wedge j+\eta \leq n \\ \{(1), (2), \dots, (j)\}, & \text{if } j-\eta < 1 \wedge j+\eta \leq n \\ \{(1), \dots, (j+\eta-n)\} \cup \{(j-\eta), \dots, (j)\}, & \text{otherwise.} \end{cases}$$

and

$$N_i^+(k) = \begin{cases} \{(j), (j+1), \dots, (j+\eta)\}, & \text{if } j+\eta \leq n \wedge j-\eta \geq 1 \\ \{(j), (j+1), \dots, (n)\}, & \text{if } j+\eta > n \wedge j-\eta \geq 1 \\ \{(n+j-\eta), \dots, (n)\} \cup \{(j), \dots, (j+\eta)\}, & \text{otherwise.} \end{cases}$$

and define the set of all neighbors  $N_i(k) := N_i^-(k) \cup N_i^+(k)$ .

□

The nearest circular value enforces all nodes to establish  $2\eta$  links, as shown in Figure 3.1d. In a social context, this definition amounts to a node with a strong opinion complementing it with some nodes with the opposite opinion, as an attempt to increase the convergence rate. Notice that this is unlikely to happen naturally in a social network, but could be enforced by policies or in scenarios where agents are given incentives to cooperate. This type of rule is often used in public debates where people with a wide range of opinions are asked to share their views on a topic of interest.

In Figure 3.1, each policy is depicted to highlight the differences in the network topology of each definition. After introducing the stochastic version of these networks in the next section, we will be focusing on providing convergence rate results and on the final opinion of the agents of the network. Both topics are of interest for example, in a company environment where one might need to arrange teams to discuss a topic or in manipulating the final opinion in an advertisement campaign.

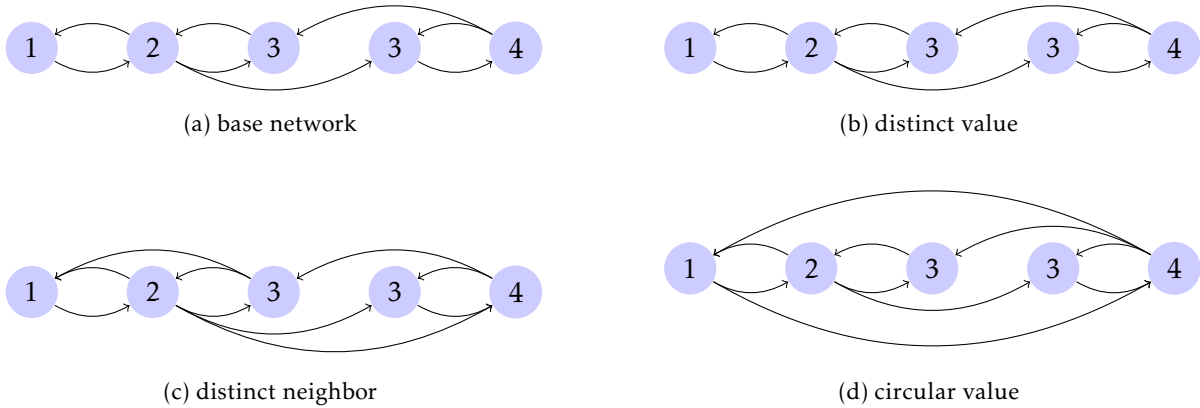


Figure 3.1: Network generated for each definition using  $\eta = 1$  and  $x_1 = 1, x_2 = 2, x_3 = 3, x_4 = 3$  and  $x_5 = 4$ .

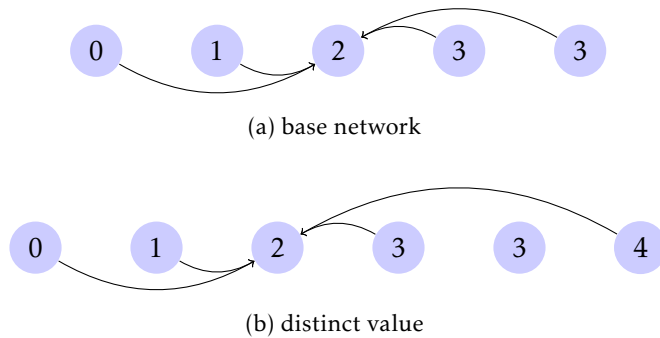


Figure 3.2: Detail of the links from node  $x_3$  when using  $\eta = 2$  and  $x_1 = 0, x_2 = 1, x_3 = 2, x_4 = 3, x_5 = 3$  and  $x_6 = 4$  for the Base and distinct value networks.

### 3.5 Stochastic State-Dependent Social Network

In this section, we introduce a randomized version of the social network presented in Section 3.4. Intuitively, at each discrete time instant, one agent is selected randomly according to the probabilities in the matrix

$$P = \begin{bmatrix} p_1 & 0 & \cdots & 0 \\ 0 & p_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & p_n \end{bmatrix}$$

where each  $p_\ell \in (0, 1)$  represents the probability that agent  $\ell$  is selected, with  $\sum_\ell p_\ell = 1$ . We denote by  $i_k$  the random variable accounting for the selection of the node updating its state at communication time  $k$ . All random variables  $i_k$  are independent and identically distributed (i.i.d.), following the distribution given by matrix  $P$ , i.e.,  $i_k = \ell$  with probability  $p_\ell$ . If a given agent  $\ell$  is selected at time  $k$ ,  $i_k = \ell$ , then its state is updated according to the update law in (3.1), but the states of all remaining agents stays unchanged.

Parameter  $\alpha_k$  is assumed to be randomly selected at each time instant  $k$  from a probability distribution with  $\bar{\alpha} := \mathbb{E}[\alpha_k]$ ,  $\forall k \geq 0$  and support  $[0, 1]$ . This definition is assuming implicitly that the distribution for the choice of  $\alpha$  is the same at every time instant, independent across time, and is common to all the nodes in the network. From the definition of the  $\alpha$  parameter, we also have that  $0 \leq \bar{\alpha} \leq 1$ . All the random variables are measurable on the same probability space  $(\Omega, \mathcal{F}, \mathbb{P})$ .

For stochastic social networks, we consider the following convergence definition to a final opinion  $x_\infty(\omega) := c(\omega)\mathbf{1}_n$ , for some constant  $c$  that depends on the outcome  $\omega \in \Omega$  encompassing the outcomes of the random variables  $\alpha_k$  and  $i_k$ .

**Definition 3.6.** *We say that the social network with graph dynamics as in Section 3.4 and stochastic selection of agents converges, in the mean square sense, to a final opinion, if there exists a random variable, given the outcome  $\omega$ , of the form  $x_\infty(\omega) := c(\omega)\mathbf{1}_n$  such that*

$$\lim_{k \rightarrow \infty} \mathbb{E}[\|x(k, \omega) - x_\infty(\omega)\|^2] \rightarrow 0.$$

An alternative to the dynamics considered above is also studied which we refer in the sequel to as “random neighbors social network”. The selection of updating node is maintained, using the random variables  $i_k$  to represent the node  $i$  selected at time  $k$ , and  $\alpha_k$  as the random choice for the parameter to use in (3.1). However, at time  $k$ , the selection of neighbors ignores the previous definitions for the connectivity graph. Instead, a set of neighbors is selected at random with equal probability from the all possible non-empty subsets constructed using the nodes in  $\mathcal{V}$ . In addition, it is made the union of the selected set with the node  $i_k$  itself, as to reflect that it is always possible for node  $i_k$  to use its own opinion. As a consequence, (3.1) is still well-defined. Let us also define the random variables  $j_k$ , as the node with minimum opinion from the selected set of neighbors at time  $k$ , and, conversely,  $\ell_k$  as the node with the maximum opinion at time  $k$ .

The random neighbors social network mimics the behavior of interaction where nodes just randomly encounter others and the stochastic updates follow the asynchronous setting of the real world. As an example of how nodes interact, consider a 6-node network with initial state  $\begin{bmatrix} 1 & 3 & 20 & -4 & 7 & 0 \end{bmatrix}^T$ , where  $i_k = 1$  and node 1 selects nodes 2, 3 and 6 to update its opinion. This would mean that  $x_1(k+1) = \alpha_k x_6(k) + (1 - \alpha_k)x_3(k)$ .

In the next section, we look at convergence rates and final opinion, both for the deterministic and the stochastic setting of all the network dynamics that we are analyzing. Whenever relevant, we will draw attention to the connection between these results and others in the literature.

### 3.6 Main Properties

#### 3.6.1 Deterministic Social Network

This section is devoted to the derivation of the convergence properties of the base social network dynamics with particular focus on the conditions to achieve finite-time convergence. The same analysis is also performed for the three policies introduced as “rules” to get a faster convergence in a social network about a given topic.

The two following Lemmas are straightforward to deduce and we present them here to simplify the subsequent proofs of convergence for social networks with the four graph dynamics.

**Lemma 3.1** (order preservation). *Take any two nodes  $i, j \in \mathcal{V}$  with the update rule (3.1) and graph dynamics described either by Definition 3.2, Definition 3.3, or Definition 3.4. If  $x_i(k) \leq x_j(k)$  for some  $k$ , then  $x_i(k+1) \leq x_j(k+1)$ .*

*Proof.* The lemma results from the relationship that if  $x_i(k) \leq x_j(k)$ , then

$$\min_{\ell \in \mathcal{N}_i(k)} x_\ell(k) \leq \min_{m \in \mathcal{N}_j(k)} x_m(k)$$

and also

$$\max_{\ell \in \mathcal{N}_i(k)} x_\ell(k) \leq \max_{m \in \mathcal{N}_j(k)} x_m(k)$$

and since the update (3.1) performs a weighted average between minimum and maximum opinions, the conclusion follows.  $\square$

Notice that Lemma 3.1 is not valid for the case of the nearest circular value of Definition 3.5, as nodes interact with neighbors that are the “farthest”. The result can be interpreted as each agent knowledge of advantages and disadvantages remain ordered as nodes contact with closer-in-opinion neighbors who in turn interact with other nodes with knowledge of more extreme facts about the topic in discussion. However, Lemma 3.1 is only used to prove asymptotic convergence, whereas a different technique is used when addressing finite-time convergence, which is the relevant result for Definition 3.5. Lemma 3.1 ensures that the relative order of the states will remain constant along time. The result will be helpful since, in the analysis, we can use the numbering of each node to sort their beliefs.



**Lemma 3.2** (convergence for higher connectivity). *Take any of the network dynamics in Definition 3.2, Definition 3.3, or Definition 3.4, and two integers  $1 \leq \eta_1 \leq \eta_2$ . Define*

$$V^\eta(k) := \max_{i \in \mathcal{V}} x_i^\eta(k) - \min_{i \in \mathcal{V}} x_i^\eta(k)$$

where  $x_i^\eta(k)$  represents the state at time instant  $k$  evolving according to (3.1) when the maximum number of larger or smaller neighbors is  $\eta$ . Then, for any initial conditions  $x(0)$ ,  $V^{\eta_1}(k) \geq V^{\eta_2}(k)$ .

*Proof.* Regardless of the value of  $\eta$  and given the iteration in (3.1), any element of  $x(k+1)$  is going to be a weighted average of the elements in  $x(k)$  with weights  $\alpha_k$  and  $1 - \alpha_k$ . Applying (3.1) recursively yields that any opinion is going to be a weighted average of the initial state with weights being all the combinations from  $\alpha_0 \cdots \alpha_k$  to  $(1 - \alpha_0) \cdots (1 - \alpha_k)$ . If we use a binary vector  $b$  to generate all the weights, it means that each combination from  $\alpha_0 \cdots \alpha_k$  to  $(1 - \alpha_0) \cdots (1 - \alpha_k)$  can be written as:

$$\prod_{i=1}^k b_i \alpha_{i-1} + (1 - b_i)(1 - \alpha_{i-1})$$

for each binary vector  $b \in \{0, 1\}^k$ .

In addition, iteration (3.1) is going to perform a weighted average of two other nodes that depend on which network dynamics is selected. Following this, we can define a function  $\varphi(i, b, k, \eta)$  used to determine the indices of the nodes selected for the average at node  $i$ , corresponding to the weight combination  $b$  and for  $k$  time instants after the initial time using a connectivity parameter  $\eta$ . For the example of the base network, going from  $k-1$  to  $k$  means that this function whether selects node  $i + \eta$  (the weight corresponds to the maximum node in (3.1)) or  $i - \eta$  (the weight corresponds to the minimum node in (3.1)). Since a node index cannot be smaller than 1 or higher than  $n$ , the  $\varphi$  function should saturate for each recursive iteration in  $k$ .

Using these two facts enables rewriting  $V^\eta(k)$  as a function of the initial state  $x(0) = x^{\eta_1}(0) = x^{\eta_2}(0)$  as:

$$V^\eta(k) = \sum_{b \in \{0, 1\}^k} \left[ \prod_{i=1}^k b_i \alpha_{i-1} + (1 - b_i)(1 - \alpha_{i-1}) \right] \left[ x_{\varphi(n, b, k, \eta)}(0) - x_{\varphi(1, b, k, \eta)}(0) \right] \quad (3.2)$$

where

$$\varphi(c, b, \ell, \eta) = \begin{cases} \text{sat}(c + (-1)^{b_1} \eta), & \text{if } \ell = 1 \\ \varphi(\text{sat}(c + (-1)^{b_\ell} \eta), b, \ell - 1, \eta), & \text{otherwise} \end{cases}$$

using the saturation function

$$\text{sat}(c) = \begin{cases} 1, & \text{if } c \leq 1 \\ n, & \text{if } c \geq n \\ c, & \text{otherwise.} \end{cases}$$

The presented function  $\varphi(\cdot)$  is for Definition 3.2 and similar functions can be given for the remaining definitions of network dynamics by adding to  $\eta$  the number of nodes with equal state.

Nevertheless, the important feature of this function is stated next and is sufficient for proving the result.

The form in (3.2) means that  $V^{\eta_1}(k)$  and  $V^{\eta_2}(k)$  represent a sum of terms multiplied by weights that are equal. Even though the weight associated with a given  $x_i(0)$  state might be different in  $V^{\eta_1}(k)$  and  $V^{\eta_2}(k)$ , the approach herein is to directly compare each term  $x_{\varphi(n,b,k,\eta)}(0) - x_{\varphi(1,b,k,\eta)}(0)$  for the two values  $\eta_1$  and  $\eta_2$ , since the weight that multiplies each of these terms is independent of  $\eta$ .

Assuming the labeling of the nodes as the relative ordering at the initial state, to prove  $x_{\varphi(n,b,k,\eta_1)}(0) \geq x_{\varphi(n,b,k,\eta_2)}(0)$  it is only required to show the equivalent for the indices, i.e.,  $\varphi(n,b,k,\eta_1) \geq \varphi(n,b,k,\eta_2)$ . Given the recursive definition of  $\varphi(\cdot)$ , one can prove by induction that for any  $k$  the inequality holds.

Let us start with the base case of  $k = 1$  and prove that  $\varphi(n,b,1,\eta_2) \leq \varphi(n,b,1,\eta_1)$ . If  $b_1 = 0$ , we have  $\varphi(n,b,1,\eta_1) = \varphi(n,b,1,\eta_2) = n$ . When  $b_1 = 1$ ,  $\varphi(n,b,1,\eta_1) = n - \eta_1$  and  $\varphi(n,b,1,\eta_2) = n - \eta_2$ . Since  $\eta_2 \geq \eta_1$ , we have proved the base case.

Consider the induction hypothesis that  $\varphi(n,b,k,\eta_2) \leq \varphi(n,b,k,\eta_1)$ . To prove the inductive step, assume  $b_{k+1} = 0$  and we have  $\varphi(n,b,k+1,\eta_2) = \varphi(n,b,k,\eta_2)$  and also  $\varphi(n,b,k+1,\eta_1) = \varphi(n,b,k,\eta_1)$ , which resorting to the induction hypothesis proves the inductive step for  $b_{k+1} = 0$ . When  $b_{k+1} = 1$ , applying the definition of the  $\varphi(\cdot)$  function asserts that  $\varphi(n,b,k+1,\eta_2) = \varphi(n-\eta_2,b,k,\eta_2)$  and  $\varphi(n,b,k+1,\eta_1) = \varphi(n-\eta_1,b,k,\eta_1)$ . We also have that  $\varphi(c,b,k,\eta) \geq \varphi(c_1,b,k,\eta)$  if  $c \geq c_1$  which allows to write

$$\begin{aligned} \varphi(n,b,k+1,\eta_2) &= \varphi(n-\eta_2,b,k,\eta_2) \\ &\leq \varphi(n-\eta_1,b,k,\eta_2) \\ &\leq \varphi(n-\eta_1,b,k,\eta_1) \\ &= \varphi(n,b,k+1,\eta_1). \end{aligned}$$

A similar proof can be obtained for  $\varphi(1,b,k,\eta)$  and therefore we have the following inequalities:

$$x_{\varphi(n,b,k,\eta_1)}(0) \geq x_{\varphi(n,b,k,\eta_2)}(0)$$

and

$$x_{\varphi(1,b,k,\eta_1)}(0) \leq x_{\varphi(1,b,k,\eta_2)}(0)$$

which implies that each term in the summation in (3.2) for  $\eta_1$  is going to be greater than or equal to the same term in (3.2) for  $\eta_2$ , thus implying the conclusion. Notice that the relationship above for  $\varphi(\cdot)$  is valid for Definitions 3.2, 3.3 and 3.4.  $\square$

#### 3.6.2 Base Network

The next theorem, which can be seen as a generalization of [SJ13], presents convergence results for the base social network.

**Theorem 3.1.** Consider a social network as in Definition 3.2 with update rule (3.1) and any sequence  $\{\alpha_k\}$ . Then,

- (i) If  $\eta \geq n - 1$ , the network is guaranteed to have finite-time convergence;
- (ii) If  $\eta < n - 1$ , the network achieves at least asymptotic convergence.

*Proof.* (i) The proof is straightforward by noticing that for  $\eta = n - 1$ , we get a complete graph and finite-time convergence is achieved in one time instant for any sequence  $\{\alpha_k\}$ .

(ii) We start by considering the case of  $\eta = 1$ . Take nodes  $i$  and  $j$  to be, respectively, the nodes with the smallest and largest states. Then,  $V^1(0) = x_j^1(0) - x_i^1(0) > 0$ , unless  $x_i^1(k) = x_j^1(k)$  (in which case convergence has already been achieved), and, thus, from the definition of the dynamics in (3.1)

$$\begin{aligned} x_i^1(k) &\leq x_i^1(k+1), \\ x_j^1(k) &\geq x_j^1(k+1). \end{aligned}$$

The important step here is to notice that at least one of the conditions must be a strict inequality. Equality only happens when  $\alpha_k = 0$  or  $\alpha_k = 1$  for at most one of the inequalities since the set  $\mathcal{N}_i^+(k) \setminus \{i\} \neq \emptyset$  and  $\mathcal{N}_j^-(k) \setminus \{j\} \neq \emptyset$ . These sets are empty only for the trivial case of having a network composed of a single node in which case, consensus is already achieved. In the first case of  $\alpha_k = 0$ ,  $x_i^1(k) < x_i^1(k+1)$  since the smallest state is subject to a maximization with a greater value. The converse is also true for the case of  $\alpha_k = 1$ . Thus,  $V^1(k+1) < V^1(k)$  which means that the sequence  $V^1(k)$  is monotonically decreasing. In addition,  $V^1(k) > 0$  except when  $x^1(k) = c1_n$  for some constant  $c$ , since by definition the neighbor set will be given by  $\mathcal{N}_i(k) = \{i\}$ . Using (3.1), we get  $x_i^1(k+1) = x_i^1(k)$  and  $V^1(k+1) = V^1(k)$ . By the discrete-time version of the La Salle Invariance Principle, the conclusion follows. Due to Lemma 3.2, since  $V^1(k)$  converges, so does  $V^\eta(k)$ , which concludes the proof.  $\square$

**Remark 3.1** (Distinct state values). In any of the graph dynamics considered in this chapter, two nodes with the same state value are not neighbors following the definitions (essentially since they are not going to affect one another). In addition, any two nodes  $i$  and  $j$  with the same state value have  $\mathcal{N}_i(k) = \mathcal{N}_j(k)$ ,  $\forall k \geq 0$ . Thus, the cardinality of the set of (distinct) node values

$$\Phi(k) = |\{x_1(k), \dots, x_n(k)\}|$$

is a non-increasing function. Moreover, if the initial states are not distinct, then the conclusions of all theorems and propositions in this section will hold, but replacing  $n$  by  $n - \Phi(0)$ . Also notice that, in the previous theorem, if  $\alpha_k = 0$  or  $\alpha_k = 1$ , then  $\Phi(k+1) = \Phi(k) - 1$ , which means that, after  $n$  time instants, convergence is achieved.

The following proposition provides the convergence rate for the case of asymptotic convergence in Theorem 3.1 when the sequence of  $\alpha_k$  is constant.

### Chapter 3: Randomized State-Dependent Algorithms

---

**Proposition 3.1.** *Consider a social network as in Definition 3.2 with update rule (3.1) and distinct initial condition  $x(0)$ , a constant sequence  $\{\alpha\}$  and  $\eta < n - 1$ . Then, the following inequality holds true*

$$x(k) - x_\infty \leq \lambda_2^k (x(0) - x_\infty)$$

where  $\lambda_2$  is the second largest eigenvalue of matrix  $A \in \mathbb{R}^{n \times n}$  defined by

$$[A]_{ij} := \begin{cases} \alpha, & \text{if } j = \max(1, i - \eta) \\ 1 - \alpha, & \text{if } j = \min(n, i + \eta) \\ 0, & \text{otherwise} \end{cases}$$

*Proof.* Since  $\alpha$  is constant and by Lemma 3.1 the ordering of the nodes does not change, we can write the state-dependent network dynamics as a state-independent iteration. Matrix  $A$ , representing one iteration of the social network, where we assumed the labeling of the nodes corresponds to the ordering of the initial states, can be used to define a linear time-invariant dynamic system written as

$$x(k+1) = Ax(k).$$

Given that  $A$  is row stochastic, it has one eigenvector  $\mathbf{1}_n$  corresponding to the eigenvalue 1 and  $x_\infty = c\mathbf{1}_n$  for some constant  $c$  defining the final social opinion. This eigenvalue has multiplicity one since  $A$  is irreducible and aperiodic given that the network is strongly connected with two self-loops by definition. Therefore,  $x_\infty = Ax_\infty$  and we can rewrite

$$\begin{aligned} x(k+1) - x_\infty &= A(x(k) - x_\infty) \\ &= A^k(x(0) - x_\infty). \end{aligned}$$

The convergence speed is governed by the magnitude of the second largest eigenvalue, thus leading to the conclusion.  $\square$

The next theorem provides a result for the base social network which is based on the eigenvectors of a matrix representing the interaction in a time step.

**Theorem 3.2** (Base Network Final Opinion). *Consider a social network as in Definition 3.2 with  $n$  nodes with distinct initial condition  $x_i(0), 1 \leq i \leq n$  and a constant parameter  $\alpha$  in (3.1). The final opinion of the network is given by*

$$x_\infty = \frac{\mathbf{1}_n w_1^\top}{\sqrt{n}} x(0)$$

where  $w_1$  is the normalized left-eigenvector associated with the eigenvalue 1 of matrix  $A \in \mathbb{R}^{n \times n}$  defined by

$$[A]_{ij} := \begin{cases} \alpha, & \text{if } j = \max(1, i - \eta) \\ 1 - \alpha, & \text{if } j = \min(n, i + \eta) \\ 0, & \text{otherwise} \end{cases}$$

*Proof.* An iteration for the base social network as in Definition 3.2 is described by matrix  $A$  when labeling the nodes according to their relative ordering, which, by Lemma 3.1, remains constant, leading to the linear description  $x(k+1) = Ax(k)$ . Thus,

$$\begin{aligned} x_\infty &= \lim_{k \rightarrow \infty} x(k) \\ &= \lim_{k \rightarrow \infty} A^k x(0). \end{aligned}$$

Notice that matrix  $A$  is row stochastic, so the eigenvalue 1 has corresponding right eigenvector  $\frac{1_n}{\sqrt{n}}$  and all the remaining eigenvalues have magnitude smaller than 1. Therefore,

$$\lim_{k \rightarrow \infty} A^k = \frac{1_n w_1^\top}{\sqrt{n}}$$

which concludes the proof.  $\square$

**Remark 3.2** (symmetric case). Assuming that  $\alpha = 0.5$  and  $\eta = 1$ , the final opinion is given by

$$x_\infty = \frac{1_n 1_n^\top}{n} x(0),$$

as matrix  $A$  is symmetric and becomes doubly stochastic, resulting in the left eigenvector to become  $w_1 = \frac{1_n}{\sqrt{n}}$ . Therefore, appropriately selecting the parameters of a social network can render the nodes to converge to the average of their positions.

An interesting remark regarding Theorem 3.2 is the appearance of  $w_1$  in the expression for the final value to which the network converges, which is the so-called *PageRank* for matrix  $A$  [IT10]. This connection comes from the fact that the base social network, for constant parameter  $\alpha$ , becomes a linear iteration for a fixed network structure.

Theorem 3.2 also relates the importance of the nodes based on the left-eigenvector, which is a centrality measure for this network (see [Fri91] for a connection between centrality measures and social networks). In this section, we will also show that this measure changes drastically depending on the chosen network dynamics.

In the proof of Theorem 3.2, we only require i) the ordering of the nodes to remain constant, which is ensured by Lemma 3.1; ii) the matrix  $A$  to be constant, which is valid for all cases when only asymptotic convergence is achieved and  $\alpha$  is constant. This allows us to introduce the behavior of the system in the presence of social leaders, i.e., nodes that do not change their opinion and serve the purpose of driving the general opinion towards a given value.

**Proposition 3.2.** Consider a social network as in Definition 3.2 and update rule (3.1) with  $n$  nodes, with a subset  $\{\ell_m\}$  with  $m = 1, 2, \dots, \mu$  of leaders, such that  $x_j(k) = x_j(0), \forall k$  if  $j \in \{\ell_m\}$ , distinct initial conditions  $x_i(0), 1 \leq i \leq n$  and a constant parameter  $\alpha$ . The network opinion converges to

$$x_\infty = \sum_{i=1}^{\mu} v_i w_i^\top x(0)$$

where  $v_i$  and  $w_i$  are, respectively, the right and left mutually orthonormal eigenvectors associated with eigenvalue 1 of matrix  $A \in \mathbb{R}^{n \times n}$ , defined by

$$[A]_{ij} := \begin{cases} 1, & \text{if } j = i \wedge i \in \ell_m \\ \alpha, & \text{if } j = \max(1, i - \eta) \wedge i \notin \ell_m \\ 1 - \alpha, & \text{if } j = \min(n, i + \eta) \wedge i \notin \ell_m \\ 0, & \text{otherwise} \end{cases}.$$

*Proof.* The proof follows by applying Theorem 3.2 and noticing that, since the left and right eigenvectors are mutually orthonormal, we have that  $\forall j \neq i : v_i^\top w_j = 0$  and  $\forall j = i : v_i^\top w_j = 1$ .  $\square$

Proposition 3.2 can be applied to all network dynamics for which Lemma 3.1 holds and for constant sequences of  $\alpha$ , by appropriately defining matrix  $A$  to that of a single iteration of the network. Proposition 3.2 holds because, in the presence of leaders, no finite-convergence is achieved, unless all nodes start with the same opinion.

#### 3.6.3 Nearest Distinct Values

In a realistic scenario, the theorem for the base network (Theorem 3.1) states that finite-time convergence of all the agents cannot be guaranteed unless  $\eta = n - 1$ , which corresponds to all nodes communicating with each other. We note that this requirement is a consequence of a poor neighbor selection, since there are unnecessary interactions with agents with the same argument. The next theorem shows the convergence results when the graph dynamics is described as in Definition 3.3, where we use the ceiling operator  $\lceil \cdot \rceil$  to denote the smallest integer greater or equal than the argument.

**Theorem 3.3.** *Consider the social network as defined in Section 3.3.3, with the graph dynamics as in Definition 3.3, and any sequence  $\{\alpha_k\}$ . Then,*

- (i) *If  $\eta \geq \frac{n}{2}$ , the network is guaranteed to have finite-time convergence in no more than  $\lceil \log_2 n \rceil$  steps;*
- (ii) *If  $\eta < \frac{n}{2}$ , the network achieves at least asymptotic convergence.*

*Proof.* (i) Without loss of generality, we assume  $n = 2\eta$ , the initial states are all distinct as in Remark 3.1, and that the numbers of the nodes are sorted according to their state ordering, so as to shorten the notation by identifying the minimum and maximum value nodes with  $x_1$  and  $x_n$ , respectively. Since  $n = 2\eta$ , there exist at least two nodes reaching the minimum and maximum nodes, exemplified in Figure 3.3b, i.e., there are  $i, j$ :

$$\begin{aligned} \min_{\ell \in \mathcal{N}_i(0)} x_\ell(0) &= \min_{\ell \in \mathcal{N}_j(0)} x_\ell(0) = x_1(0) \\ \max_{\ell \in \mathcal{N}_i(0)} x_\ell(0) &= \max_{\ell \in \mathcal{N}_j(0)} x_\ell(0) = x_n(0) \end{aligned}$$

Thus,  $\Phi(1) = \Phi(0) - 1$ . In the subsequent iterations the cardinality reduces by  $2, 4, \dots$  by nodes fulfilling the previous conditions, which leads to  $\Phi_k = n - (2^k - 1)$ . Hence,  $\Phi(k) \leq 1 \Leftrightarrow k \geq \log_2 n$ , thus leading to the conclusion.

(ii) Using the previous argument, one determines that if  $\eta < \frac{n}{2}$ , it is not possible to find at least a pair of nodes communicating with the whole network and guarantee finite-time convergence. Asymptotic convergence is achieved by the argument in the proof of Theorem 3.1 and by noticing that the graph dynamics in Definition 3.3 also imply a strongly connected graph with at least one self-loop.  $\square$

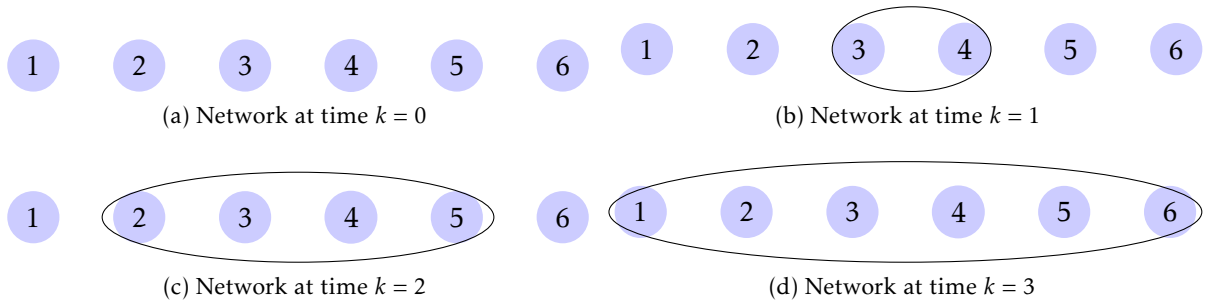


Figure 3.3: Convergence of the social network under the policy of distinct value and considering  $n = 6$  and  $\eta = \frac{n}{2}$  where the ellipses represent a cluster of nodes with equal opinions.

The convergence described in the proof is illustrated in Figure 3.3. As mentioned, when  $k = 1$  as in Figure 3.3b the two nodes with median opinions have access to the whole network and, thus, they form a cluster of beliefs. In Figure 3.3d, it is not captured the exponential behavior due to limited space to represent a network with more nodes.

Theorem 3.2 provides a categorization of the final opinion for the base social network which depends on a left eigenvector of a matrix, but it is not straightforward to understand how the steady state is influenced by the initial conditions. In the sequel, closed-form results are presented that describe the dependence on the initial conditions when finite-time convergence is achieved for the network dynamics as in Definition 3.3 and Definition 3.4. The case of distinct values is presented next.

**Theorem 3.4.** Consider a social network with dynamics as described in Definition 3.3 and distinct initial conditions  $x_i(0), 1 \leq i \leq n$ , with parameters  $\alpha = \frac{1}{2}$  and  $\eta = \lceil \frac{n}{2} \rceil$ . The network opinion converges to

$$x_\infty = \frac{\Gamma}{2^{\lceil \log_2 n \rceil}} \mathbf{1}_n$$

where

$$\Gamma = \sum_{j=1}^{\lceil \log_2 n \rceil} \lceil 2^{\lceil \log_2 n \rceil - 1 - j} \rceil (x_{1+\theta_j} + x_{n-\theta_j})$$

using the following definitions for the indices

$$\theta_j = \begin{cases} 0, & \text{if } j = 1 \\ \sum_{i=1}^{j-2} [(-1)^{i+1} \Phi(i)] + \eta, & \text{if even } j \\ \sum_{i=1}^{j-1} [(-1)^{i+1} \Phi(i)] - 1, & \text{if odd } j > 1 \end{cases}$$

where, recall that  $\Phi(k) := |\{x_1(k), \dots, x_n(k)\}|$ .

*Proof.* We start our proof by showing that  $\theta$  is the set of indices of the initial states that contribute to the final opinion value. At time instant  $k = 1$ , the minimum node will have a state equal to the weighted average between  $x_1$  (i.e., the node with minimum state at time  $k = 0$ ) and  $x_{1+\eta}$  and, conversely, the maximum state will be the weighted average between  $x_n$  and  $x_{n-\eta}$ , thus obtaining the second term  $\eta$ .

In the next time instant, the minimum value node contacts the node that is the  $\eta$ -th smaller value which corresponds to adding the node  $x_{1+(1+2\eta) \bmod n} = x_{1+n-\Phi(1)}$  and conversely to the maximum value getting  $x_{\Phi(1)}$ . The key aspect to notice is that  $\Phi(1)$  was added to take into account that the cardinality of nodes with distinct values has decreased. By following the same pattern, we obtain the expression for  $\theta$ .

To finalize the proof, we must compute the weights associated with each index. We notice that the aggregation is a binary tree and the weights double after each time instant that the index was added to  $\theta$ . Thus, the weights are given by  $2^{\lceil \log_2 n \rceil - 1 - j}$  where we must subtract 1 since the time starts at  $k = 0$  and  $j$  accounts for the time instant it enters in the index set  $\theta$ .  $\square$

To illustrate Theorem 3.4, consider a network with  $n = 16$  and  $\eta = 8$  for  $\alpha = 0.5$  where the aim is to compute the final social opinion. Using Theorem 3.4, the final state is given by

$$x_\infty = \frac{4x_1 + x_2 + x_6 + 2x_8 + 2x_9 + x_{11} + x_{15} + 4x_{16}}{32} \mathbf{1}_n$$

while if  $\eta = n - 1$  the solution is

$$x_\infty = \frac{x_1 + x_{16}}{2} \mathbf{1}_n,$$

which indicates that the minimum and maximum opinion nodes are the most influential in the final network belief and as  $\eta$  increases their preponderance follows.

#### 3.6.4 Nearest Circular Value

The next theorem presents the convergence results when the graph dynamics are as in Definition 3.5.

**Theorem 3.5.** *Consider the social network with graph dynamics as in Definition 3.5, update rule (3.1) and any sequence  $\{\alpha_k\}$ . Then, for any  $\eta \geq 1$ , the network has finite-time convergence in no more than  $\lceil \frac{n-(2\eta+1)}{2\eta-1} \rceil + 1$  time steps.*



*Proof.* Without loss of generality, we assume distinct initial states as in Remark 3.1 and that the nodes labels are sorted according to their state ordering. If  $\Phi(k) \leq 2\eta + 1$ , then we have the complete network and finite-time consensus is achieved in a single time instant.

At each time  $k$ , there are  $2\eta$  nodes that have access both to  $x_1(k)$  and  $x_n(k)$ . Thus,  $\Phi(k) = n - (2\eta - 1)k$  and we need to have  $\Phi(k) \leq 2\eta + 1 \Leftrightarrow k \geq \frac{n-(2\eta+1)}{2\eta-1}$  to get to a configuration where finite-time convergence is achieved in a single time instant, which concludes the proof.  $\square$

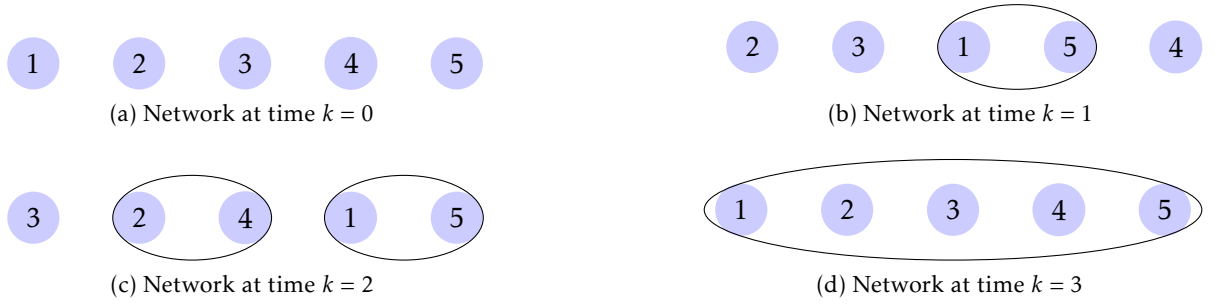


Figure 3.4: Convergence of the social network under the policy of circular value and considering  $n = 5$  and  $\eta = 1$ , where the ellipses represent a cluster of nodes with equal opinions.

The convergence of the previous policy is depicted in Figure 3.4. The nodes are numbered in their initial ordering according to their state values to indicate that such is not maintained between iterations. The relative position where the cluster forms is not meaningful to the convergence and was selected arbitrarily in Figure 3.4b and Figure 3.4c.

**Remark 3.3.** In a first analysis, the convergence time provided in Theorem 3.3, i.e.,  $\log_2 n$ , could appear significantly faster when compared to  $\lceil \frac{n-(2\eta+1)}{2\eta-1} \rceil + 1$  from Theorem 3.5. However, we stress that, in Theorem 3.3, such a rate is achieved when  $n = 2\eta$ , which would lead to convergence in a single instant in the conditions of Theorem 3.5.

### 3.6.5 Nearest Distinct Neighbors

The following result shows convergence for the case when the network graph dynamics is as in Definition 3.4.

**Theorem 3.6.** Consider the social network with graph dynamics as in Definition 3.4, update rule (3.1) and any sequence  $\{\alpha_k\}$ . Then, for any  $\eta \geq 1$ , the network has finite-time convergence in no more than  $\lceil \frac{n-(2\eta+1)}{2\eta} \rceil + 1$  time steps.

*Proof.* Without loss of generality, we assume distinct initial states as in Remark 3.1 and that the nodes labels are sorted according to their state ordering. Similarly to the previous theorem, if  $\Phi(k) \leq 2\eta + 1$  then the network is complete between all the nodes with distinct values and finite-time consensus is achieved in a single time instant.

At each time  $k$ , there are  $\eta + 1$  nodes that have access to  $x_1(k)$  and  $x_{1+\eta}(k)$ , and  $\eta + 1$  nodes receive the information  $x_{n-\eta}(k)$  and  $x_n(k)$ . Thus,  $\Phi(k) = n - 2\eta k$  and we need to have  $\Phi(k) \leq 2\eta + 1 \Leftrightarrow k \geq \frac{n-(2\eta+1)}{2\eta}$  to get to a configuration where finite-time convergence is achieved in a single instant, which concludes the proof.  $\square$

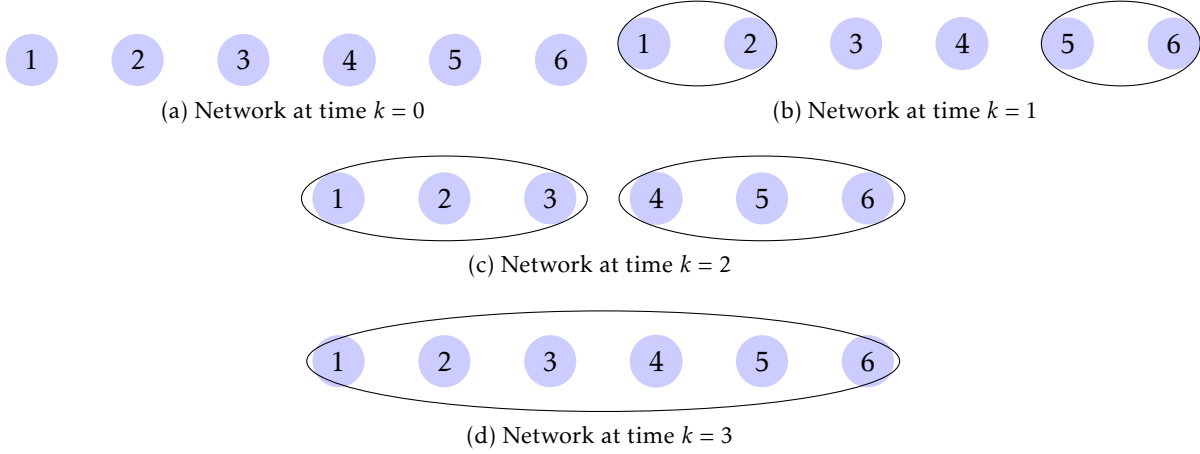


Figure 3.5: Convergence of the social network under the policy of Distinct Neighbor and considering  $n = 6$  and  $\eta = 1$ , where the ellipses represent a cluster of nodes with equal opinions.

Figure 3.5 illustrates how the social network converges with the network dynamics as in the previous theorem. The formation of two clusters in Figure 3.5b and then its enlargement in each of the subsequent iterations (as illustrated in Figure 3.5c) is the main idea of this policy.

The clustering behavior observed when using the previous definition for selecting neighbors is very different from what is obtained when using Definition 3.4, where the median nodes play the key role. The result is summarized in the following theorem.

**Theorem 3.7.** Consider a social network with graph dynamics as in Definition 3.4, update rule (3.1), and distinct initial conditions  $x_i(0), 1 \leq i \leq n$ , with parameter  $\alpha = \frac{1}{2}$ . The network opinion when  $n = 1 + 2\eta\ell, \ell = \{0, 1, \dots\}$  converges to

$$x_\infty = \frac{\sum_{j=0}^{\tau} \binom{\tau}{j} x_{1+j2\eta}}{2^\tau} \mathbf{1}_n \quad (3.3)$$

where  $\tau = \lceil \frac{n}{2\eta} \rceil - 1$ . For the remaining values of  $n$  we get

$$x_\infty = \frac{\sum_{j=0}^{\tau} \binom{\tau}{j} [x_{1+j2\eta} + x_{n-(\tau-j)2\eta}]}{2^{\tau+1}} \mathbf{1}_n. \quad (3.4)$$

$\square$

*Proof.* For the trivial case of  $n = 2$ , the expression is straightforward to verify. For the general case, we use induction to prove the result. Start by noticing that for  $k = 1$ , we get weighted

averages of pairs of variables of the form  $x_i + x_{i+2\eta}$ . When  $k = 2$ , the averages are of the type  $x_i + 2x_{i+\eta} + x_{i+2\eta}$  since  $n > 2\eta$ , or otherwise an additional communication step would not be necessary and  $k$  would be one.

Using the previous observation, we need to consider 3 cases: when  $n = 1 + 2\eta\ell$ , when  $1 + 2\eta\ell < n < 2\eta(\ell + 1)$ , and when  $n = 2\eta\ell$ .

(i) When  $n = 1 + 2\eta\ell$ , there exists an instant  $k$  such that  $\Phi(k - 1) = 2\eta + 1$ . For time instant  $k$ ,  $\Phi(k) = 1$  and for  $n$  nodes, by assumption, all of their values is a weighted average in the form of (3.3) for time  $k$ , i.e.,

$$\frac{\sum_{j=0}^k \binom{k}{j} x_{1+j2\eta}}{2^k}. \quad (3.5)$$

If we consider  $n + 1$ , from the previous observation, there will be a node at time  $k$  with the value of

$$\frac{\sum_{j=0}^k \binom{k}{j} x_{2+j2\eta}}{2^k}$$

and where the last term of the sum is, by definition, dependent on  $x_n$ . Thus, we can rewrite it as

$$\frac{\sum_{j=0}^k \binom{k}{j} x_{n-(k-j)2\eta}}{2^k}. \quad (3.6)$$

By combining equations (3.5) and (3.6), we get that all nodes at time  $k + 1$  achieve (3.4).

(ii) For this case, the proof is similar to the previous one by noting that, since  $n < 2\eta(\ell + 1)$ , at time  $k - 1$ ,  $\Phi(k - 1) < 2\eta$  for the case of  $n$  nodes. Thus, when considering  $n + 1$ , the same setting as before is achieved.

(iii) When  $n = 2\eta\ell$ , we get that at time  $k - 1$ ,  $\Phi(k - 1) = 2\eta$ . When considering the case of  $n + 1$ , we will get at time  $k$  exactly  $2\eta + 1$  distinct values with the minimum

$$\frac{\sum_{j=0}^k \binom{k}{j} x_{1+j2\eta}}{2^k} \quad (3.7)$$

and maximum

$$\frac{\sum_{j=0}^k \binom{k}{j} x_{1+(1+j)2\eta}}{2^k}. \quad (3.8)$$

Since the last element of equation (3.8) must be  $x_n$  by definition, we can rewrite the equation as to count the variables from  $n$  instead of 1 and get

$$\frac{\sum_{j=0}^k \binom{k}{j} x_{n-j2\eta}}{2^k}. \quad (3.9)$$


---

### Chapter 3: Randomized State-Dependent Algorithms

---

Combining equations (3.7) and (3.9), noticing that all terms are repeated except for the first term in (3.7) and last term in (3.9), and given the fact that

$$\binom{k}{j} = \binom{k-1}{j-1} + \binom{k-1}{j},$$

the social network for  $n+1$  final value is as in equation (3.3) when considering that it takes an additional time instant to converge.  $\square$

As a small example to illustrate Theorem 3.7, let us consider a network with  $n = 16$  and  $\eta = 2$ . Hence, one obtains

$$x_\infty = \frac{x_1 + x_4 + 3x_5 + 3x_8 + 3x_9 + 3x_{12} + x_{13} + x_{16}}{16} \mathbf{1}_n$$

which shows that under the network dynamics of Definition 3.4, the most influential nodes are close to the median and not the minimum and maximum nodes, as in the case of Definition 3.3. These results sustain the fact that given different objectives, it might be beneficial to choose one network over the other and scale the connectivity parameter  $\eta$  accordingly.

#### 3.6.6 Stochastic Social Network

Section 3.5 introduced the stochastic version of the social network presented in this chapter, which relaxes the condition that all the nodes are influenced deterministically at the same time instants. By considering the stochastic model of the network, we allow for the asynchronous case to be considered, which is closer to the actual dynamics that we are trying to model. We start by analyzing the case where the network dynamics is the base version and the case where nodes only select distinct opinions, in the following theorem.

**Theorem 3.8.** *Consider a stochastic social network with graph dynamics as in Definition 3.2 or as in Definition 3.3 with connectivity parameter  $\eta$ , update rule (3.1), and initial conditions  $x_i(0), 1 \leq i \leq n$ , with parameter  $\alpha_k$  following a probability distribution with mean  $\bar{\alpha}$ . Then, the network opinion converges to a consensus in the mean square sense.*

$\square$

In the proof, all inequalities and equalities involving random variables are valid for a arbitrary  $\omega \in \Omega$  and occur with probability one.

*Proof.* We start by defining the shorter notation for the minimum and maximum as

$$x_{\min}(k, \omega) := \min_{\ell} x_{\ell}(k, \omega)$$

$$x_{\max}(k, \omega) := \max_{\ell} x_{\ell}(k, \omega)$$

and the limit random variable  $c(\omega)$ , for an outcome  $\omega$  of the random selections  $i_k$  and all random variables  $\alpha_k$ , is defined as

$$c(\omega) := \lim_{k \rightarrow \infty} x_{\min}(k, \omega)$$

which exists and is measurable by the Monotone Convergence Theorem, since  $x_{\min}(k, \omega)$  is a monotonically increasing sequence and upper bounded by  $x_{\max}(0)$  for all outcomes  $\omega$ .

Also, given the definition of the function  $V^\eta(\cdot)$  in Lemma 3.2,  $\forall k \geq 0$

$$\begin{aligned} \|x(k, \omega) - x_\infty(\omega)\|^2 &= \sum_{\ell=1}^n (x_\ell(k, \omega) - c(\omega))^2 \\ &\leq V^\eta(x(0)) \sum_{\ell=1}^n |x_\ell(k, \omega) - c(\omega)| \\ &\leq V^\eta(x(0)) \sum_{\ell=1}^n x_{\max}(k, \omega) - x_{\min}(k, \omega) \end{aligned} \quad (3.10)$$

where the inequalities in (3.10) are given by the relationship  $\forall \ell \in \mathcal{V}, k \geq 0 : |x_\ell(k, \omega) - c(\omega)| \leq x_{\max}(k, \omega) - x_{\min}(k, \omega)$ , which comes directly from the definition of minimum and maximum. Note that the updating rule in (3.1) performs convex combinations, i.e.,  $x_\ell(k+1, \omega) = \sum_{q=1}^n a_q x_q(k, \omega)$  for some weights  $a_q$  with  $\sum_{q=1}^n a_q = 1$ . Therefore,  $x_{\min}(k, \omega)$  and  $x_{\max}(k, \omega)$  are respectively monotonically increasing and decreasing and  $\forall \ell \in \mathcal{V}, k \geq 0 : |x_\ell(k, \omega) - c(\omega)| \leq x_{\max}(0) - x_{\min}(0)$  since  $\forall k \geq 0 : x_{\max}(k, \omega) \leq x_{\max}(0)$  and  $\forall k \geq 0 : x_{\min}(k, \omega) \leq x_{\min}(0)$ .

Using (3.10), it follows

$$\mathbb{E}[\|x(k, \omega) - x_\infty(\omega)\|^2 | x(0)] \leq V^\eta(x(0)) \mathbb{E}[V^\eta(x(k, \omega)) | x(0)].$$

We shall prove, for  $\eta = 1$ , that

$$\mathbb{E}[V^\eta(x(k, \omega)) | x(0)] \leq \bar{\gamma}^k V^\eta(x(0)) \quad (3.11)$$

from which stability in the mean square sense follows, because

$$\mathbb{E}[\|x(k, \omega) - x_\infty(\omega)\|^2 | x(0)] \leq \rho \bar{\gamma}^k V^\eta(x(0))^2.$$

for some positive constant  $\rho$  and  $\bar{\gamma} < 1$ .

Let us start with  $\eta = 1$ . In this case, when  $\alpha_k \in (0, 1)$ , since  $\eta = 1$ , we can take the labeling of the nodes to be their relative order such that  $x_1(0) \leq x_2(0) \leq \dots \leq x_n(0)$ . This labeling is not changed since  $\forall \ell \in \mathcal{V} \setminus \{1, n\}, k \geq 0 : x_{\ell-1}(k, \omega) \leq x_\ell(k, \omega) \leq x_{\ell+1}(k, \omega)$  due to  $x_\ell(k, \omega)$  being a convex combination of  $x_{\ell-1}(k-1, \omega)$  and  $x_{\ell+1}(k-1, \omega)$ . For the nodes with the minimum and maximum state, the converse is true, i.e.,  $\forall k \geq 0 : x_1(k, \omega) \leq x_2(k, \omega)$  and  $\forall k \geq 0 : x_{n-1}(k, \omega) \leq x_n(k, \omega)$ . When considering some  $\alpha_k = 0$  or  $\alpha_k = 1$ , one can take the relative order of the nodes at time  $k$  instead of their labeling, i.e., replace 1 by (1), 2 by (2), and conversely for the remaining nodes for all the expressions of this proof.

From the previous observation, the random variable  $x(k, \omega)$  takes the form of a linear system of the type  $x(k+1, \omega) = Q_{i_k}(\alpha_k)x(k, \omega)$ , where matrices  $Q_i(\alpha)$  are defined as

$$[Q_i(\alpha)]_{j\ell} := \begin{cases} \alpha, & \text{if } \ell = \max(1, j-1) \wedge j = i \\ 1 - \alpha, & \text{if } \ell = \min(n, j+1) \wedge j = i \\ 1, & \text{if } j = \ell \wedge j \neq i \\ 0, & \text{otherwise.} \end{cases}$$

### Chapter 3: Randomized State-Dependent Algorithms

for nodes  $i, j, \ell \in \mathcal{V}$  and a parameter  $\alpha \in [0, 1]$ . Matrices  $Q_i(\alpha)$  are equivalent to taking row  $i$  from matrix  $A$ , defined for the deterministic in Proposition 3.1, and all the other rows from the identity matrix.

To prove (3.11) for  $\eta = 1$ , it is sufficient to show that

$$\mathbb{E}[V^1(x(k + \tau, \omega)) | x(k, \omega)] - \gamma V^1(x(k, \omega)) \leq 0 \quad (3.12)$$

for time interval of size  $\tau$ , constant  $\gamma < 1$ , which relates to  $\bar{\gamma}$  through  $\gamma^{\frac{k}{\tau}} = \bar{\gamma}^k$ , and where  $\mathbb{E}[\cdot]$  is the conditional expected value operator.

In order to upperbound the expected value in (3.12), notice by the definition of  $V^1(\cdot)$ , for all time instant  $k$ ,  $\exists i^* < j^* : x_{j^*}(k, \omega) - x_{i^*}(k, \omega) \geq \frac{V^1(x(k, \omega))}{n}$ . In particular, there exists adjacent nodes  $i^*$  and  $j^*$ , i.e.,  $j^* = i^* + 1$ . Thus,  $i^*$  and  $j^*$  cannot be 1 and  $n$  at the same time. Assuming  $i^*$  and  $j^*$  are both different from  $n$ , we can define a finite sequence  $\rho$ , of size  $\tau$ , such that  $\rho_1 = i_{k+1}, \dots, \rho_\tau = i_{k+\tau}$ . With the objective of writing  $x_1(k + \tau, \omega)$  with terms that include both  $x_1(k, \omega)$  and  $x_n(k, \omega)$ , we consider the finite sequence  $\rho_1^* = n - 1, \rho_2^* = n - 2, \dots, \rho_\tau^* = 1$ . This sequence of updates, of size  $\tau = n - 1$  occurs with non-zero probability

$$p_{\text{good}} = \prod_{\ell=1}^{\tau} [P]_{\ell}.$$

Computing the product  $Q_1(\alpha_{k+\tau-1}) \dots Q_{n-1}(\alpha_k) x(k, \omega)$  allows to write the expected value of function  $V^1(\cdot)$  subject to the chosen sequence  $\rho^*$  to occur from time  $k$  to  $k + \tau$  as

$$\begin{aligned} \mathbb{E}[V^1(x(k + \tau, \omega)) | x(k, \omega), \rho = \rho^*] &= x_n(k, \omega) - \mathbb{E}[(\alpha_{k+\tau-1} + \alpha_{k+\tau-2}(1 - \alpha_{k+\tau-1}))x_1(k, \omega) | x(k, \omega)] \\ &\quad - \mathbb{E}\left[\sum_{\ell=2}^{\tau-1} \left(\alpha_{k+\tau-\ell-1} \prod_{j=0}^{\ell-1} 1 - \alpha_{k+\tau-j-1}\right) x_{\ell}(k, \omega) | x(k, \omega)\right] \\ &\quad - \mathbb{E}\left[\left(\prod_{\ell=0}^{\tau-1} (1 - \alpha_{k+\ell})\right) x_n(k, \omega) | x(k, \omega)\right] \end{aligned} \quad (3.13)$$

where the conditional expected values in (3.13) are over the random variables  $\alpha_k, \alpha_{k+1}, \dots, \alpha_{k+\tau-1}$ . Since  $\alpha_k$  is assumed to be independently selected in each time instant  $k$ ,  $\forall k \geq 0, \phi > 0$  :  $\mathbb{E}[\alpha_k \alpha_{k+\phi}] = \mathbb{E}[\alpha_k] \mathbb{E}[\alpha_{k+\phi}]$ . Thus, and due to linearity of the expected value operator, (3.13) can be simplified to

$$\begin{aligned} \mathbb{E}[V^1(x(k + \tau, \omega)) | x(k, \omega), \rho = \rho^*] &= x_n(k, \omega) - \\ &\quad \left[ \bar{\alpha}(2 - \bar{\alpha})x_1(k, \omega) + \sum_{\ell=2}^{\tau-1} (\bar{\alpha}(1 - \bar{\alpha})^{\ell})x_{\ell}(k, \omega) + (1 - \bar{\alpha})^{\tau}x_n(k, \omega) \right]. \end{aligned} \quad (3.14)$$

Lastly, due to the fact that the nodes labeling correspond to their relative ordering, we can upperbound (3.14) and get

$$\begin{aligned} \mathbb{E}[V^1(x(k + \tau, \omega)) | x(k, \omega), \rho = \rho^*] &\leq x_n(k, \omega) - [(1 - (1 - \bar{\alpha})^{\tau})x_1(k, \omega) + (1 - \bar{\alpha})^{\tau}x_n(k, \omega)] \\ &\leq (1 - (1 - \bar{\alpha})^{\tau})(x_n(k, \omega) - x_1(k, \omega)) \\ &\leq (1 - (1 - \bar{\alpha})^{\tau})V^1(x(k, \omega)) \end{aligned} \quad (3.15)$$

where all the  $x_\ell(k, \omega)$  inside the summation in (3.14) were replaced by  $x_1(k, \omega)$ . Remark that

$$\begin{aligned}\mathbb{E}[V^1(x(k+\tau, \omega))|x(k, \omega)] &= \sum_{\rho} p_{\rho} \mathbb{E}[V^1(x(k+\tau, \omega))|x(k, \omega), \rho] \\ &= p_{\text{good}} \mathbb{E}[V^1(x(k+\tau, \omega))|x(k, \omega), \rho = \rho^*] \\ &\quad + \sum_{\rho_s \neq \rho^*} p_{\rho_s} \mathbb{E}[V^1(x(k+\tau, \omega))|x(k, \omega), \rho = \rho_s]\end{aligned}$$

where  $p_{\rho}$  is the probability of occurring the finite sequence  $\rho$  out of all possible finite sequences of size  $\tau$ . Given the upperbound in (3.15) for the chosen sequence and that for all the remaining  $\rho_s, \forall k \geq 0, \tau \geq 0 : V^1(x(k+\tau, \omega)) \leq V^1(x(k, \omega))$ , the expected value in (3.12) can be upperbounded by

$$\mathbb{E}[V^1(x(k+\tau, \omega))|x(k, \omega)] \leq p_{\text{good}} (1 - (1 - \bar{\alpha})^{\tau}) V^1(x(k, \omega)) + (1 - p_{\text{good}}) V^1(x(k, \omega)) \quad (3.16)$$

By simplifying (3.16), we get

$$\mathbb{E}[V^1(x(k+\tau, \omega))|x(k, \omega)] \leq [1 - p_{\text{good}}(1 - \bar{\alpha})^{\tau}] V^1(x(k, \omega))$$

which satisfies (3.12) for  $\gamma = 1 - p_{\text{good}}(1 - \bar{\alpha})^{\tau}$ .

For the other case where  $i^*$  and  $j^*$  are both different from 1, following a similar reasoning, we would select the finite sequence  $\rho_1^* = 2, \rho_2^* = 3, \dots, \rho_{\tau}^* = n$ . Following the same steps would lead to

$$\mathbb{E}[V^1(x(k+\tau, \omega))|x(k, \omega)] \leq [1 - p_{\text{good}} \bar{\alpha}^{\tau}] V^1(x(k, \omega))$$

which satisfies (3.12) for  $\gamma = 1 - p_{\text{good}} \bar{\alpha}^{\tau}$ . Inequality (3.12) holds for both cases by selecting  $\gamma = 1 - p_{\text{good}} \max(\bar{\alpha}^{\tau}, (1 - \bar{\alpha})^{\tau}) < 1$  which confirms that (3.11) holds for  $\eta = 1$ , from which convergence in mean square sense follows for  $\eta = 1$ .

As for  $\eta > 1$ , applying the same reasoning as in the proof of Lemma 3.2, we have

$$0 \leq V^{\eta}(x(k, \omega)) \leq V^1(x(k, \omega)) \quad (3.17)$$

which means that for a generic  $\eta$ , the function  $V^{\eta}(\cdot)$  is upperbounded by  $V^1(\cdot)$ . Combining (3.11) and (3.17) leads to

$$\mathbb{E}[V^{\eta}(x(k, \omega))|x(0)] \leq \bar{\gamma}^k V^1(x(0))$$

from which convergence in mean square follows for  $\eta > 1$ , thus concluding the proof.  $\square$

In the next theorem, we analyze the convergence for the case of distinct neighbors.

**Theorem 3.9.** *Consider a stochastic social network with graph dynamics as in Definition 3.4, update rule (3.1) and initial conditions  $x_i(0), 1 \leq i \leq n$  with parameter  $\alpha_k$  following a probabilistic distribution with mean  $\bar{\alpha}$ . The, the network opinion converges to a consensus in mean square sense.*

$\square$

### Chapter 3: Randomized State-Dependent Algorithms

*Proof.* The proof follows a similar reasoning as that of Theorem 3.8 and focus on establishing (3.12). Similarly to Theorem 3.8, taking  $\eta = 1$  makes possible to write the random variable  $x(k, \omega)$  in the form of a linear system of the type  $x(k+1, \omega) = Q_i(\alpha_k)x(k, \omega)$ , but with matrices  $Q_i(\alpha)$  being defined as

$$[Q_i(\alpha)]_{j\ell} := \begin{cases} \alpha, & \text{if } \ell = \max(1, \min(j-1, n-2)) \wedge j = i \\ 1 - \alpha, & \text{if } \ell = \min(n, \max(j+1, 3)) \wedge j = i \\ 1, & \text{if } j = \ell \wedge j \neq i \\ 0, & \text{otherwise.} \end{cases}$$

for nodes  $i, j, \ell \in \mathcal{V}$  and  $\alpha \in [0, 1]$ . Matrices  $Q_i(\alpha)$  are equivalent to taking row  $i$  from the matrix defining the network dynamics in Definition 3.4 of the deterministic case, and all the other rows are taken from the identity matrix.

For  $\eta = 1$  and  $i^*$  and  $j^*$  both different from  $n$ , we can select  $\rho^*$  of length  $\tau = n - 2$  such that  $\rho_1^* = n - 1, \rho_2^* = n - 2, \dots, \rho_{\tau-1}^* = 3, \rho_\tau^* = 1$  since the update of node 2 is irrelevant due to node 1 having as neighbor both node 2 and 3 for  $\eta = 1$ . In doing so, (3.13) becomes

$$\begin{aligned} \mathbb{E}[V^1(x(k+\tau, \omega)|x(k, \omega), \rho = \rho^*)] &= x_n(k, \omega) - \mathbb{E}[\alpha_{k+\tau-1}x_1(k, \omega)|x(k, \omega)] \\ &\quad - \mathbb{E}\left[\sum_{\ell=2}^{\tau} \left( \alpha_{k+\tau-\ell} \prod_{j=0}^{\ell-2} 1 - \alpha_{k+\tau-j-1} \right) x_\ell(k, \omega) | x(k, \omega)\right] \\ &\quad - \mathbb{E}\left[\left(\prod_{\ell=0}^{\tau-1} (1 - \alpha_{k+\ell})\right) x_n(k, \omega) | x(k, \omega)\right]. \end{aligned}$$

Following that, equation (3.14) becomes

$$\mathbb{E}[V^1(x(k+\tau, \omega)|x(k, \omega), \rho = \rho^*)] = x_n(k, \omega) - \left[ \bar{\alpha}x_1(k, \omega) + \sum_{\ell=2}^{\tau} \left( \bar{\alpha}(1 - \bar{\alpha})^{\ell-1} \right) x_\ell(k, \omega) + (1 - \bar{\alpha})^\tau x_n(k, \omega) \right].$$

However, by replacing all  $x_\ell(k, \omega)$  inside the summation by  $x_1(k, \omega)$ , we get the same expression for (3.15) but with  $\tau = n - 2$  instead of  $n - 1$ . Following the same steps for  $i^*$  and  $j^*$  both different from 1 would lead to the same expression as in Theorem 3.8. Thus, by following the remaining steps in the proof of Theorem 3.8, the conclusion follows.  $\square$

Another interesting case of the stochastic social network is the random neighbors version, which is analyzed in the next theorem.

**Theorem 3.10.** *Consider a random neighbors social network and initial conditions  $x_i(0), 1 \leq i \leq n$ , with parameter  $\alpha_k$  following a probabilistic distribution with mean  $\bar{\alpha}$ . Then, the network opinion converges in mean square sense to consensus.*

$\square$



*Proof.* Let us recall the random variables  $i_k$  to represent the node whose clock ticked and is going to update its state and define the random variables  $j_k$  as the minimum node selected by node  $i_k$  at time  $k$ , and  $\ell_k$  as the maximum node selected by node  $i_k$  at time  $k$ . The social network takes the form of a linear system of the type  $x(k+1) = Q_{j_k \ell_k}^{i_k}(\alpha_k)x(k)$ , where matrices  $Q_{j\ell}^i(\alpha)$  are defined as

$$[Q_{j\ell}^i(\alpha)]_{qr} := \begin{cases} \alpha, & \text{if } q = i \wedge r = j \\ 1 - \alpha, & \text{if } q = i \wedge r = \ell \\ 1, & \text{if } q \neq i \wedge q = r \\ 0, & \text{otherwise.} \end{cases}$$

for nodes  $i, j, \ell, q, r \in \mathcal{V}$  and  $\alpha \in [0, 1]$  as the parameter for (3.1). In the remainder of the proof, we will omit the dependence of  $x(\cdot)$  on  $\omega$  to shorten the notation and all inequalities and equalities involving random variables hold for an arbitrary  $\omega$  with probability one.

Let us compute the probabilities associated with each of the matrices  $Q_{j\ell}^i(\cdot)$  for a given value of  $i, j$  and  $\ell$ . Let us define matrices  $\Pi_i$ , where  $[\Pi_i]_{j\ell}$  is the probability that after selecting node  $i$  its update uses the minimum as node  $j$  and the maximum as node  $\ell$

$$[\Pi_i]_{j\ell} := \begin{cases} \frac{2^{\ell-j}}{2^n-1}, & \text{if } j = i \wedge j \leq \ell \\ \frac{2^{\ell-j-1}}{2^n-1}, & \text{if } j < i \wedge i < \ell \\ \frac{2^{\ell-j}}{2^n-1}, & \text{if } j < i \wedge i = \ell \\ 0, & \text{otherwise.} \end{cases}$$

The probability of each  $Q_{j\ell}^i(\alpha)$  is going to be the probabilities in  $[\Pi_i]_{j\ell}$  multiplied by the probability distribution function of  $\alpha$ . Let us also define matrix

$$R = \mathbb{E}[Q_{j\ell}^i(\alpha)].$$

Then,

$$\mathbb{E}[x(k+1)] = R^k \mathbb{E}[x(0)]$$

due to the probability distribution of selecting each matrix  $Q_{j\ell}^i(\alpha)$  and the corresponding parameter  $\alpha$  being independent. The expected value matrix  $R$  can be written as

$$R = \frac{1}{n} \left( (n-1)I + (1 - \bar{\alpha})(I \otimes \mathbf{1}_n^T)\Omega + \bar{\alpha}\Upsilon \right)$$

where

$$\Omega := \begin{bmatrix} \Pi_1 \\ \Pi_2 \\ \vdots \\ \Pi_n \end{bmatrix}$$

and

$$[\Upsilon]_{ij} := \begin{cases} \frac{2^{n-j}}{2^n-1}, & \text{if } i > j \\ \frac{2^{n-j+1}-1}{2^n-1}, & \text{if } i = j \\ 0, & \text{otherwise.} \end{cases}$$

### Chapter 3: Randomized State-Dependent Algorithms

Matrices  $\Pi_i$  have all entries summing to 1, making each  $1_n^\top \Pi_i$  sums to 1, leading to  $(I \otimes 1_n^\top) \Omega$  being row stochastic and upper triangular. In addition, matrix  $\Upsilon$  is also row stochastic but lower triangular. As a consequence, matrix  $R$  is a full matrix with all positive entries and row stochastic as it is a convex combination of row stochastic matrices. Thus, according to the Gershgorin's disk theorem, it has all eigenvalues within the unit circle. Since  $R$  is full, it is irreducible and, by the Perron-Frobenius theorem, it only has one eigenvalue equal to 1, showing that the limit of the expected value converges. These properties are required for the proof of convergence in the mean square sense.

Similarly, let us introduce the matrix

$$R_2 := \sum_{i=1}^n \sum_{j=1}^n \sum_{\ell=1}^n [\Pi_i]_{j\ell} Q_{j\ell}^i \otimes Q_{j\ell}^i.$$

Manipulating the expression, and given that the distributions are independent, we can write

$$\mathbb{E}[x(k+1)x(k+1)^\top] = R_2^k \mathbb{E}[x(0)x(0)^\top].$$

Due to the structure of matrices  $Q_{j\ell}^i$ , the second moment matrix can be written as

$$\begin{bmatrix} \Gamma_1 & \Lambda_{1,2} & \Lambda_{1,3} & \cdots & \Lambda_{1,n} \\ \Lambda_{2,1} & \Gamma_2 & \Lambda_{2,3} & \cdots & \Lambda_{2,n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \Lambda_{n-1,1} & \Lambda_{n-1,2} & \cdots & \Gamma_{n-1} & \Lambda_{n-1,n} \\ \Lambda_{n,1} & \Lambda_{n,2} & \cdots & \Lambda_{n,n-1} & \Gamma_n \end{bmatrix}$$

where

$$\Gamma_\ell = R - \sum_{i \neq \ell} \left[ (1 - \bar{\alpha}) [\Pi_i]_{j\ell} Q_{\ell i}^\ell + \bar{\alpha} [\Pi_i]_{j\ell} Q_{i\ell}^\ell \right] - \sum_{i \neq \ell} \sum_{j \neq \ell, j \neq i} Q_{ij}^\ell$$

and

$$\Lambda_{\ell j} = \sum_{i \neq j} \left[ (1 - \bar{\alpha}) [\Pi_i]_{j\ell} Q_{ij}^\ell + \bar{\alpha} [\Pi_i]_{j\ell} Q_{ji}^\ell \right].$$

Matrix  $R_2$  is still a row stochastic matrix but with non-negative entries. In order to show that  $R_2$  is irreducible, consider its support graph given by having  $n^2$  nodes corresponding to the dimension of  $R_2$  and having an edge  $(i, j)$  for each  $[R_2]_{ij} \neq 0$ . Notice that in the block diagonal, we have full matrices and, therefore, have  $n$  complete graphs of  $n$  nodes each. Since the support graph of  $\Lambda_{\ell j}$  has a link  $(\ell, j)$  which connects the  $\ell$  of one of the clusters with  $j$  of another, the overall graph is still connected. Following the same reasoning, all the eigenvalues are within the unit circle with only one eigenvalue 1 and the conclusion follows.  $\square$

The proofs regarding the convergence of the considered social network use similar steps and tools that can be used for addressing other network dynamics. However, the focus of this work is on these specific network dynamics, as they reflect the observation of social networks in real-life.

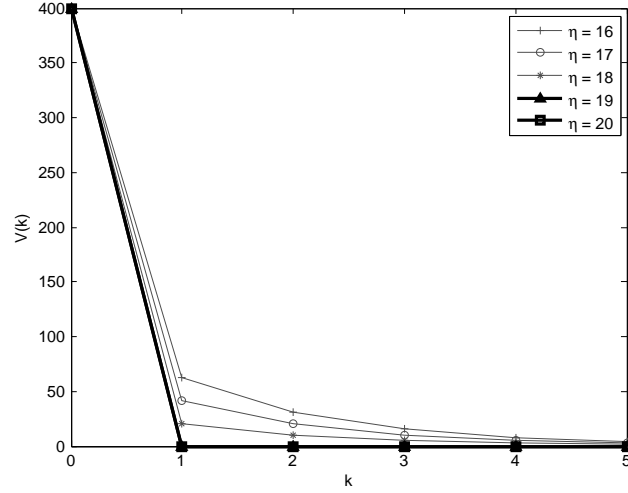


Figure 3.6: Evolution of  $V(k)$  for the case of a base social network for values of  $\eta = 16, \dots, 20$ .

### 3.7 Simulation Results

In the previous section, we showed convergence results for four different settings: what we devise as a base social network according to the observations and assumptions; a version where people contact only other agents with distinct opinions; a first strategy where people with strong beliefs search for agents with opposite arguments; and a last setting where nodes contact with exactly  $2\eta$  nodes.

In order to compare these four policies, we consider a social network with  $n = 20$  agents and set their initial states to  $x_i(0) = i^2, i = 1, \dots, n$ , and set  $\alpha_k = \frac{1}{2}, \forall k \geq 0$ .

Figure 3.6 depicts the evolution of function  $V(k)$  in each iteration of the base social network. Recall that  $V(k)$  denotes, as defined in the statement of Lemma 3.2, the distance between the largest and smallest nodes of the network. The case where  $\eta = 19$  is overlapped by the case of  $\eta = 20$  as in both cases we are dealing with the complete network where all nodes connect to the whole network.

The simulation for the case of a social network where nodes follow the distinct value policy is presented in Figure 3.7. The cases of finite-time convergence (depicted in thick lines) correspond to  $\eta \geq \frac{n}{2}$ . The maximum number of iterations corresponds to the value provided by Theorem 3.3. Whereas in the base network finite-time convergence is only guaranteed for the complete network, in this case only two nodes must receive information from the whole network.

Figures 3.8 and 3.9 show the simulation results for the circular graph dynamics and the closest distinct neighbor policy, respectively. We draw attention to the fact that both rules lead to finite-time convergence regardless of the choice of  $\eta$ , but that the closest distinct policy has a faster rate. In the circular policy, a cluster of nodes contacting the two nodes with the strongest

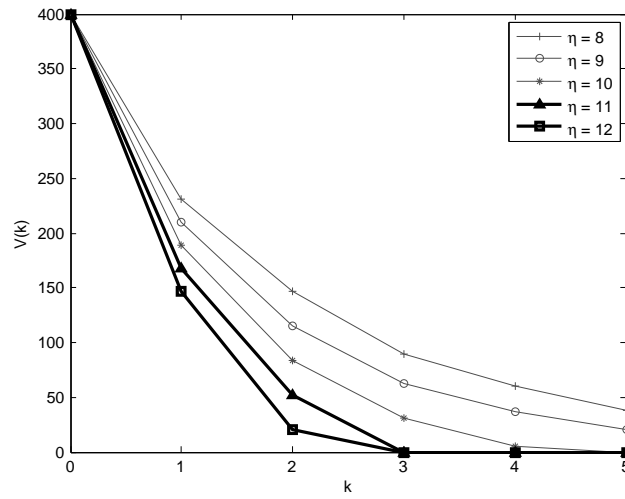


Figure 3.7: Evolution of  $V(k)$  for the case of a social network with agents communicating with nodes with distinct opinions for values of  $\eta = 8, \dots, 12$ .

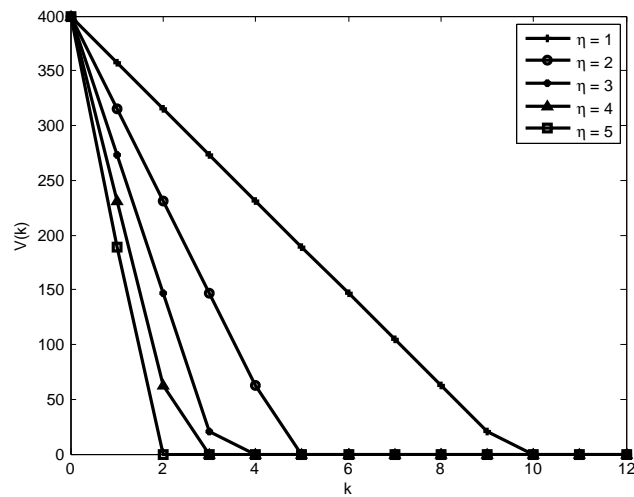


Figure 3.8: Evolution of  $V(k)$  for the case of a social network with agents with strong opinion looking for opposite opinions for values of  $\eta = 1, \dots, 5$ .

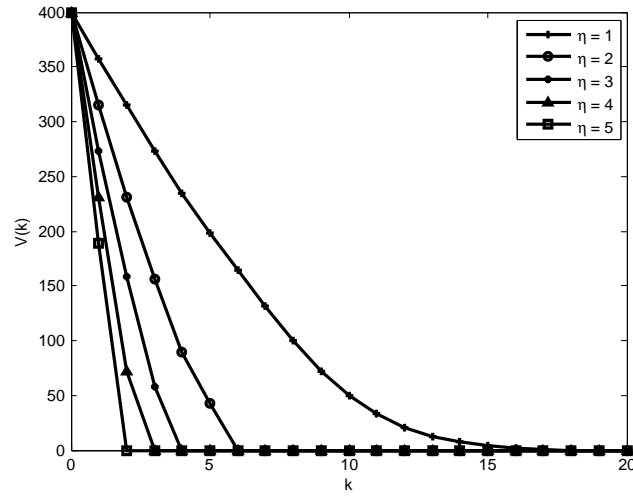


Figure 3.9: Evolution of  $V(k)$  for the case of a social network with agents contacting the  $2\eta$  closest distinct neighbors for values of  $\eta = 1, \dots, 5$ .

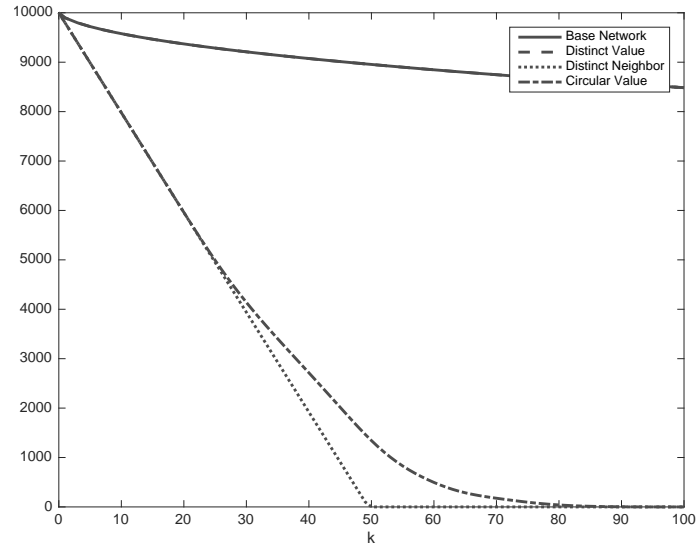


Figure 3.10: Comparison of the evolution of  $V(k)$  for the four cases with  $\eta = 1$ .

opinions is formed in each iteration. In contrast, for the closest distinct neighbor strategy, two clusters of nodes with the same opinion are formed in the first iteration and, in each subsequent steps, new nodes are added.

The previous simulations are useful to illustrate the results presented before. However, it is not straightforward to compare the convergence of all four scenarios. In a different simulation, we increase the number of nodes to  $n = 100$  and set  $\eta = 1$  to make the results comparable, since the first two scenarios have a number of links equal to  $\eta(2n - \eta - 1)$  and the following two have  $2n\eta$  links.

Figure 3.10 depicts the range of the state, as measured by the function  $V(k)$  for the different networks. Both the circular and distinct neighbor achieve finite-time convergence. The main conclusion is that both graph dynamics corresponding to Definition 3.2 and Definition 3.3, are

restrictions, leading to slow convergence rates. For the case of  $\eta = 1$ , they are the same since the lines are overlapped. We also point out the behavior of the circular and distinct neighbor policies to enforce fast convergence. This indicates that forcing the establishment of clusters of opinions leads to finite-time convergence and that the rate is governed both by the number of clusters and how fast other nodes join those clusters.

In the previous section, we showed results regarding what is the final social opinion using different network dynamics and connectivity parameters. Definition 3.5 was not addressed since the relative order of the states is not preserved, which prevents the use of our analysis. However, in our simulations, evidence supports that such a definition presents similar performance to that of Definition 3.4 for the considered cases.

In order to compare these four policies, we consider a social network with  $n = 100$  agents and three different cases for the initial conditions:

- initial conditions are drawn from independent normal distributions with expected value 100 and variance 1;
- initial states are chosen from independent exponential distributions with  $\lambda = 100$ ;
- and a final example where 90 nodes are chosen from a normal distribution with expected value 1 and 10 agents are selected from normal distributions with expected value 100.

Figure 3.11 depicts the evolution of the final opinion value  $x_\infty$  as a function of  $\eta$  when considering  $\alpha_k = \frac{1}{2}, \forall k \geq 0$  and network dynamics as in Definition 3.2. The first interesting point is that, when considering all the initial states drawn from independent normal variables with expected value equal to 100 and variance 1, the final opinion converges to the expected value. Such a result can be explained by the fact that the final belief is a convex combination of the initial states, which are normally distributed. This will be observed regardless of the network dynamics and the value taken for  $\eta$ . Moreover, the final value depends only on  $\alpha$ .

In Figure 3.12, it is shown the results for the network dynamics as in Definition 3.3 which are very similar to the ones shown in Figure 3.11, in particular, since for any value of  $\eta < \frac{n}{2}$  the final value is the same. We point out that for the geometric distribution, the social opinion is smaller than what is achieved for the Circular and Neighbor dynamics. An interesting aspect for small values of  $\eta$  is that the final value is greater than what can be achieved using other dynamics since the minimum and maximum values have a higher weight as suggested by Theorem 3.4.

Figure 3.13 depicts the final opinion of the network when using the Neighbor network dynamics. The final opinion increases with  $\eta$  except for the case of the normal distribution with expected value equal to 100. In the geometric distribution case, it is possible to achieve a higher opinion by selecting  $\eta$  close to  $n$  and a smaller value if we consider  $\eta$  close to 1. In the case where the population is divided into two groups, we see that the social opinion can approximate

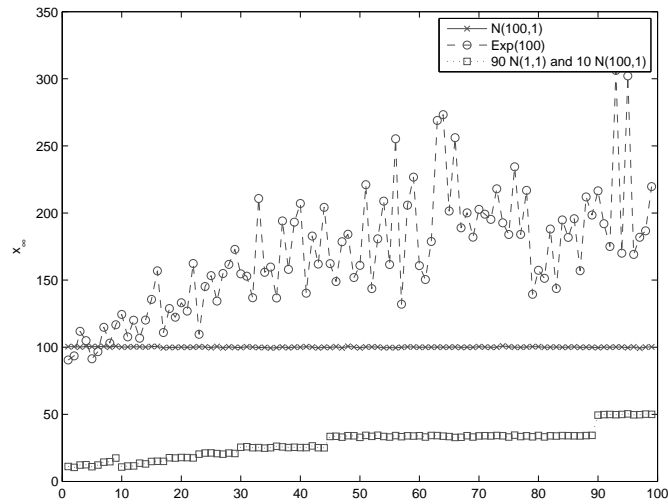


Figure 3.11: Evolution of the final state  $x_\infty$  as function of  $\eta$  for the case of the base network dynamics.

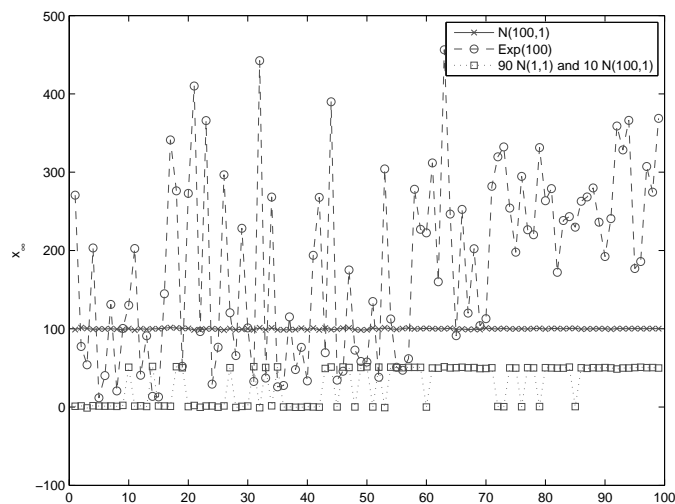


Figure 3.12: Evolution of the final state  $x_\infty$  as function of  $\eta$  for the case of the Distinct Network dynamics.

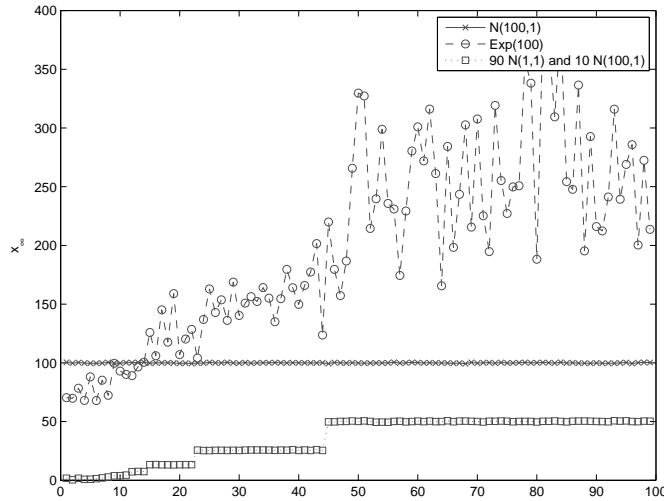


Figure 3.13: Evolution of the final state  $x_\infty$  as function of  $\eta$  for the case of the Neighbor Network dynamics.

that of the majority by selecting  $\eta = 1$ , since this policy places higher weights in the median nodes, as shown in Theorem 3.7.

As discussed in the previous section, due to the fact that the relative order of the agents is not maintained under the Circular dynamics, computing an expression for the final opinion value gets harder and would depend on the nodes initial states themselves. Intuitively, the Circular dynamics should produce results similar to those of the Neighbor policy as it had similar properties in terms of convergence, as shown in the previous section. In Figure 3.14, we depict the simulation results comparing both policies and confirm the intuition. In the cases where normal distributions were used, the final values for the Circular and Neighbor networks were different by a factor of  $10^{-2}$  except for the cases where  $\eta < 9$  in the last example, where the difference was around the order of 1. In the geometric distribution case, the social opinion converged to a higher value using the Circular definition for  $\eta < 10$ , and for  $\eta \geq 10$  becomes very similar. Thus, the simulations suggest that both policies have no substantial difference.

### 3.8 Conclusions

In this chapter, the problem of studying the evolution of the opinion in a social network associated with a political party or an association is firstly addressed using a deterministic distributed iterative algorithm with different types of graph dynamics that express how agents interact. The dynamics considered are motivated by the fact that people tend to engage discussion with those with opinions close to their own. We also consider networks exhibiting stochastic interactions between nodes.

For the deterministic setup, we show convergence results for the base social network and show that it can be improved by considering only nodes with distinct opinions. By doing so,



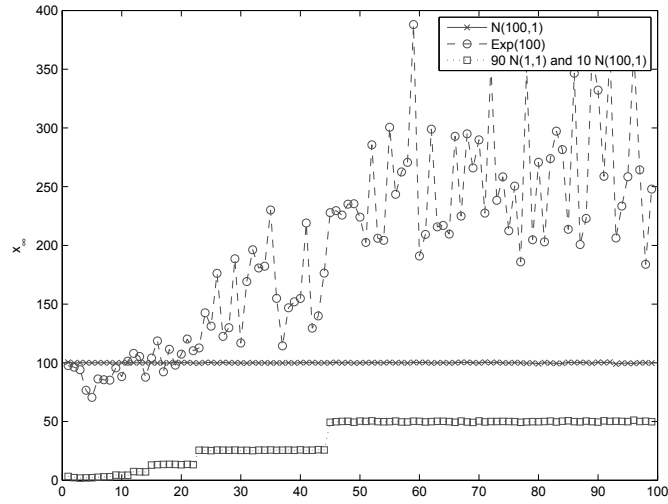


Figure 3.14: Evolution of the final state  $x_\infty$  as function of  $\eta$  for the case of the Circular Network dynamics.

convergence is attained requiring only half of the interconnections when compared to the base case. Two policies are then introduced to reduce the required parameter  $\eta$  influencing the number of interactions, namely a strategy where nodes with extreme opinions seek the influence of others with opposite argument; and a policy where agents ensure at all time  $2\eta$  links by just considering the closest agents in belief without concerns whether it is greater or smaller. Convergence results are provided that establish finite-time convergence. Both graph dynamics have different ways of creating clusters of opinions, which influences the transient behavior of the network. These results are useful in a company or organization environment, where agents can be motivated to cooperate according to one of these rules, which will attain a faster social convergence. In the stochastic setup, convergence in the mean square sense is proved for all the policies except the circular definition.

For the case of constant  $\alpha$ , we show how the final opinion depends on the left eigenvector of a row stochastic matrix representing an iteration of the algorithm. This result also applies to the case of the distinct values policy and the weights associated with each initial condition are computed when the distinct values policy converges in finite-time, showing the minimum and maximum opinion nodes are the ones having the greatest impact on the final value, followed by the nodes near the median in a logarithmic fashion.

The circular strategy does not maintain the relative order of the nodes according to their opinion. Evaluating this strategy in simulation revealed that it follows the same general behavior of the distinct neighbors policy. The distinct neighbor policy results in a social opinion where the nodes closer to the median are more influential, and the weights are given by the entries of the Pascal triangle.



# 4

## SET-VALUED ESTIMATORS

### 4.1 Introduction

In the previous chapters, the distributed algorithms were designed to deal with *crash*-like type of faults by including stochastic communications. Agents were either working correctly or non-responsive, links cannot be both working and failing during the same period, packets are either discarded or delivered, etc. In any of such cases, randomness could make the algorithm robust but is worthless if, for example, we are considering data to be corrupted in delivered packets, sensor loss of sensitivity, unmodelled interactions by third parties foreign to the algorithm, etc. The main target of this chapter is to extend the possible faults that a distributed system can detect and be robust to.

The problem of detecting faults in an asynchronous distributed environment relates to determining if any of the nodes enters in an incoherent state given the observed history of measurements. In particular, we are interested in randomized algorithms where the dynamics is common to all the nodes and no control messages are needed. This class of algorithms is used for iterative solutions because they offer a certain level of robustness against packet drops and node failure. Applications of randomized algorithms [MR10] range from computing integrals to consensus [BGPS06] and solving problems for which the solution requires a heavy computational burden [IK90] [DHKP97] [Mul94]. Large scale distributed systems and the use of robot swarms highlight the importance of this problem for practical applications.

The aim of this chapter is to detect the presence of an attacker who corrupts the states of the nodes or their transmissions. In this context, the small probability of an event cannot be discarded as an attacker can select the worst case signal, which motivates the use of set-valued estimation tools. Therefore, we address the problem in a distributed manner where each node models the network from its perspective as a Linear Parameter-Varying (LPV) system, where the input is the attacker signal. Since an attacker is allowed to inject any signal, we are looking at the worst case scenario and estimating the set of all possible state realizations that comply with

the “fault-free” model. If the set becomes empty, we can guarantee the presence of an attacker (Byzantine fault) or any other fault.

Byzantine fault detection methods have been proposed in the literature for a number of specific applications. For instance, [KMMS97] focuses on detection in the case of a consensus problem by using unreliable fault detectors, where multiple classes of theoretic detectors are presented. The proposed method checks if the algorithm is running correctly and if all the messages are in concordance with the specifications. The research interest in Byzantine faults has motivated a number of contributions including the scenario of unreliable networks in distributed systems. In particular, [PBB12] considers the problem of detecting and correcting the state of the system in the presence of a Byzantine fault. The case of malicious agents and faulty agents is studied and the authors provide, in both cases, bounds on the number of corrupted nodes to ensure detectability of the fault. In [PBB12], the system dynamics are described by a linear time-invariant model that constrains the communications in each time slot to be from a fixed set of senders to a set of receivers. Here, however, a randomized gossip algorithm is considered, thus dropping the assumption that the same set of nodes is every time involved in message exchanges.

The adopted strategy for fault detection has an interesting finite-time property that can be used in consensus problems. Finite-time consensus in the presence of malicious agents has been addressed in [SH11], where the authors show that the topology of the network categorizes its ability to deal with attacks. Both the number of corrupted nodes and vertex-disjoint paths in the network influence its resilience. In [SH11], it is assumed a broadcast model where, at each transmission time, the nodes send to all their neighbors the same value and the agents objective is to compute some function of the initial states. The main difference to the work described herein is the communications model, which we assume to be *gossip*, where pairs of nodes are selected randomly to exchange information, instead of having a broadcast model.

In [SRC<sup>+</sup>13], the concept of Stochastic Set-Valued Observers (SSVOs) was introduced by resorting to the use of  $\alpha$ -confidence sets, i.e., sets where the state of the system is guaranteed to belong with a desired pre-specified  $1 - \alpha$  probability; which can be viewed as a generalization of confidence intervals. The property of finite-time consensus when using (deterministic) Set-Valued Observers (SVOs) for a sufficiently large horizon in a randomized gossip consensus algorithm is shown in [SRHS14].

Besides the development of a theoretical framework to address the problem at hand, it is also needed to cover the mathematical machinery required to cope with the computation of the set where the current state can take values. From the random behavior of the gossip algorithm, a set-valued estimate requires the union over all possible transmission of the set of possible state realizations originated by that transmission and the previous state. By definition, the number of sets grows exponentially with the horizon  $N$ . We resort to the concept of SVOs for this task, firstly introduced in [Wit68] and [Sch68]. For the interested reader, further information can be

found in [Sch73] and [MV91] and the references therein.

An alternative to the use of SVOs is the use of zonotopes, described in [BR71] and further developed in [Com05], [ABC05] and [SRMB16]. Zonotopes represent a different trade-off between the computation complexity of unions and intersections. In particular, intersections introduce conservatism which motivated the alternative approach adopted in this chapter in order to attain the desired convergence guarantees, while keeping the computational requirements to a tractable level. The idea of interval analysis [Moo66] may also be adopted, although it introduces conservatism by not considering horizon values larger than unity in their typical formulation, unlike SVOs [RS13]. In [REZ12], interval observers for linear and nonlinear systems are proposed with mild assumptions such as the boundedness of the disturbances and measurement noise (similar to the assumptions for the SVOs).

In the literature, there are other examples of fault detection systems that employ gossip algorithms in order to achieve scalability. In [RMH98], the proposed protocol aims at detecting faults by using a gossip-like communication. The work differs from our proposal in the sense that the protocol is limited to determining unreachable nodes and does not cope well in the presence of attackers.

The applicability of the proposed method in the detection of faults in randomized gossip algorithms spans other purposes as several challenges in the Fault Detection and Isolation (FDI) literature - [Pat97, BS09] - share the framework described in the sequel. In [RSSA10], [RS13], the authors take advantage of SVOs for fault detection by resorting to a model falsification approach. This chapter extends the results in [RSSA10], [RS13] to detect Byzantine faults in randomized gossip algorithms by rewriting the associated dynamics as an LPV model. Moreover, unlike the approach in [RSSA10] and [RS13], the method proposed herein takes into account the information related to the probability of having a given communication, in order to reduce the conservatism of the results.

In [RGTC01], three algorithms are proposed for gossip-like fault detection in distributed consensus over large-scale networks, namely round-robin, binary round-robin, and round-robin with sequence check. These improve upon the basic randomized version by constructing a better gossip list and reducing the probability for false positives. The algorithms are particularly designed for the consensus problem in its version where all the nodes must select a value among the initial set of values. Our algorithm aims at detecting faults for general iterative linear distributed algorithms that can be subject to sensor noise or other effects that render the detection non-trivial.

Closely related to the concept of stochastic detection is the work presented in [RNEV08] which performs the detection by finding the change points in the correlation statistics for a sensing network. The authors are able to provide guarantees on detection delay and false alarm probability. Such approach addresses a similar problem of detecting faults that are possible in the standard dynamics but not very “probable” to take place. Our work tackles

this issue in a different way by considering the set of possible states given the more “probable” dynamics.

In the context of fault detection in distributed systems, [ME14] addresses the problem by looking at the whole system and constructing a batch of observers for each sub-system. By looking at the outputs of these observers it is possible to detect and isolate faults affecting one of the sub-systems. However, it is a centralized approach whereas our focus is to run each of the observers locally at each sub-system in a fully distributed way.

In [ZJ14], the authors propose an on-line fault detection and isolation algorithm for linear discrete-time uncertain systems where the detection is based on the computation of upper and lower bounds for the fault signal. The calculations are performed resorting to Linear Matrix Inequality (LMI) optimization techniques. Similar computational burden considerations to the work presented in this chapter are discussed and the techniques are related to our work. However, in order to address randomized gossip algorithms we studied a more general class of systems.

Using the approach of design residual filters, [CJ14] studies a class of linear continuous-time systems with the purpose of identifying faulty actuators. The aim of this work is to adjust the filters parameters as to decouple them when faults affect a group of actuators. Our approach differs in the sense that we want to incorporate unknown parameters in the dynamics matrix of the system.

### 4.2 Main Contributions and Organization

The organization of this chapter develops towards presenting all the details of fault detection for the worst-case and in the stochastic sense for distributed linear systems. Initial focus is given to distributed gossip systems and their key elements and constraints posed on the detection, namely, the characteristics associated with the network component and how faults are modeled. The concept of Set-Valued Observers (SVOs) is introduced and applied to the deterministic fault detection, as the worst-case is considered. Progress is made in presenting a method to extend the SVOs computation to incorporate the stochastic information of the communication process, which results in the Stochastic Set-Valued Observers (SSVOs).

The SVO-based fault detection method motivates the introduction of a consensus algorithm that performs averages on intervals containing the state, intersecting them upon neighbor communication. The algorithm is asymptotically convergent and also has the advantage that, under some communication patterns, it finds the consensus value in finite-time due to the intersection phase. The stochastic detection is an extension of the previous method with the set of state estimates being a subset of the previous one corresponding to a confidence set of where the state can take values. Lastly, in the particular case of consensus, it is introduced an algorithm that takes advantage of the local estimates and intersects them upon communication to generate less conservative sets. Therefore, this chapter is proposing an SVO-based approach

to fault detection with different types of SVOs. For the deterministic worst-case detection, it is proposed an SVO that can run in each node to perform fault detection using only locally available information. The stochastic detection is an extension of the previous method with the set of state estimates being a subset of the previous one corresponding to a confidence set of where the state can take values. Lastly, in the particular case of consensus, we propose an algorithm that takes advantage of the local estimates and intersects them upon communication to generate less conservative sets.

The main contributions can be found in the papers [SRC<sup>+</sup>13], [SRHS14], [SRHS15d] and [SRHS17c], and outlined as follows:

- it is shown how to compute a threshold for the “maximum impact” of an undetected fault, discussing two particular cases: linear consensus, and networked physical systems;
- the number of required communications for guaranteeing detection is reduced by analyzing the structure of randomized gossip algorithms;
- finally, we show how some of the dynamics matrices can be discarded from the model that each node has of the network, which reduces the computational complexity of the fault detection procedure.

### 4.3 Fault Detection Problem

We consider a set of  $n_x$  agents (also referred as nodes) labeled from one to  $n_x$ . Each node  $i$ , at each transmission time  $k$ , has a scalar state  $x_i(k)$ ,  $1 \leq i \leq n_x$ . At each transmission time  $k$ , each node  $i$  chooses a random out-neighbor  $j$ , according to the communication topology modeled by a connectivity graph  $G = (V, E)$ , where  $V$  represents the set of  $n_x$  agents, and  $E \subseteq V \times V$  is the set of communication links. Node  $i$  can send a message to node  $j$ , if  $(i, j) \in E$ . If there exists at least one  $i \in V$  such that  $(i, i) \in E$  we say that the graph has self-loops. By assumption, any node  $i$  has a self-loop which means that if  $i$  did not communicate, it still has access to its own value at any transmission time  $k$ . We associate to graph  $G$  a *weighted adjacency matrix*  $W$  with entries:

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

where the weight  $w_{ij} \in [0, 1]$  is the probability that node  $i$  selects  $j$  to communicate and, therefore,  $W \mathbf{1}_{n_x} = \mathbf{1}_{n_x}$ .

The “fault-free” gossip algorithm can be defined by the dynamics discrete-time equation

$$x(k+1) = A(k)x(k), \tag{4.1}$$

where the matrix  $A(k)$  is selected randomly from a set  $\{Q_{ij}, (i, j) \in E\}$ , i.e.,  $A(k) = Q_{ij}$  with probability  $\frac{w_{ij}}{n_x}$  given by the probability  $\frac{1}{n_x}$  of node  $i$  being the one initiating the communication, and probability  $w_{ij}$  of node  $j$  being selected by  $i$ . The choice for matrices  $A(k)$  model the

process by which nodes select a random out-neighbor, as described above, and where  $x(k) = [x_1(k), \dots, x_{n_x}(k)]^\top$ . Matrices  $Q_{ij}$  implement the update on state variables  $x_i$  and  $x_j$  caused by a transmission from node  $i$  to node  $j$  and represent a set of matrices that are equal to the identity except for rows  $i$  and  $j$ . In this chapter, we assume symmetry in the communication and update rule, meaning that both rows  $i$  and  $j$  are equal (which implies matrices  $A(k)$  to be symmetric), and no further structure is assumed regarding the linear iteration.

The “fault-free” algorithm in equation (4.1) is modified to include faults resulting in:

$$x(k+1) = A(k)x(k) + B(k)u(k), \quad (4.2)$$

where the input,  $u(k)$ , models the fact that some of the nodes may either report incorrect values regarding their state value or update their state by something other than the “fault-free” version. In particular, the case of an attacker trying to exploit the algorithm weaknesses motivates to consider any input signal  $u(k)$  [PBB12].

The objective of the detection algorithm is to use only limited information provided by local interactions between nodes in the network. A node performing the detection does not have access to all the communications between the remaining nodes. Indeed, the output of the system from the perspective of node  $i$ ,  $y^i(k)$ , at time  $k$ , is composed of the states that were involved in the communication with that node. In other words, if node  $j$  transmitted to node  $i$  at time  $k$ , then  $y^i(k)$  will be the vector with the states  $x_i$  and  $x_j$  i.e.,  $y^i(k) = C_i(k)x(k)$ , with  $C_i = [e_i, e_j]^\top$  and will only have its own state if the node did not communicate ( $C_i(k) = [e_i, e_i]^\top$ )<sup>1</sup>. With a slight abuse of notation, we use  $y^i(k)$  to refer to the output of the system at time  $k$  and  $y_k^i(x(0), u_k)$  to express the same output as a function of the initial state  $x(0)$  and input  $u_k$ , where  $u_k$  denotes the sequence of inputs up to time  $k$ .

The full dynamics  $S_i$  for node  $i$ , as defined above, refers to the pair of equations:

$$S_i : \begin{cases} x(k+1) = A(k)x(k) + B(k)u(k) \\ y^i(k) = C_i(k)x(k) \end{cases} \quad (4.3)$$

The main goal of this chapter can therefore be stated as: developing algorithms for detecting nonzero inputs  $u(k)$  in (4.2) that do not require knowledge of the matrices  $B(k)$ <sup>2</sup> and signal  $u(k)$  and, instead, only use the measured variables  $y_k^i$ , which stands for all the measurements up to time  $k$ , as in (4.3).

We introduce the following definition:

**Definition 4.1** (undetectable faults). *Take the randomized gossip system modeled by (4.3) from node  $i$ 's perspective. A nonzero input sequence  $u_k$  (corresponding to a fault) is said to be undetectable*

---

<sup>1</sup>Alternatively, one can consider simply  $C_i(k) = e_i^\top$ , although this would imply that the size of vector  $y^i(k)$  depends on  $k$ .

<sup>2</sup>Since the focus is on fault detection rather than fault isolation, we generate set-valued estimates for the state of the “fault-free” system which does not require the knowledge of matrices  $B(k)$



in  $N$  measurements if for some transmission sequence:

$$\forall_{k < N}, \exists_{x(0), x'(0) \in W_o} : y_k^i(x(0), u_k) = y_k^i(x'(0), 0)$$

where  $W_o$  is a set where initial state  $x(0)$  is known to belong to. Otherwise, it is said to be detectable.

□

The intuition behind this definition is that a fault is only guaranteed to be detectable if there is no possible set of initial conditions such that the sequence  $y^i(0), \dots, y^i(N)$  of measurable states can be generated without an attacker signal. The fault being detectable as in Definition 4.1 relates to the observability of the system, as described in [GG76]. Notice that if the fault does not satisfy Definition 4.1, it cannot be guaranteed its detection with probability 1. The mechanism presented throughout this chapter can still detect such faults depending on the sequence of transmissions and the initial state of the nodes.

In summary, the problem being tackled in this chapter relates to detecting any fault which cannot be generated by a “fault-free” model only with the knowledge of local measurements of the node state itself and those to which it communicates. The fault detection mechanism is distributed and no global knowledge of which nodes are communicating is assumed and neither is known the nodes or the communication links affected by the attacker.

## 4.4 Fault Detection using Set-Valued Observers (SVOs)

In this section, we analyze the fault detection problem from a deterministic point of view, and recast the network within the LPV framework. As a consequence, the random selection of matrices  $A(k)$  is disregarded and all realizations of the sequence of matrices  $A(k)$  are considered regardless of their probabilities. Firstly, we start by rewriting the matrices  $A(k)$  in (4.2) as the sum of a single central matrix  $A_0$  with parameter-dependent terms:

$$A(k) = A_0 + \sum_{\ell=1}^{n_\Delta} \Delta_\ell(k) A_\ell \quad (4.4)$$

where each  $\Delta_\ell(k)$ ,  $\forall k \geq 0$  is a scalar uncertainty with  $|\Delta_\ell(k)| \leq 1$ , and the  $A_\ell$ ,  $\ell \in \{1, 2, \dots, n_\Delta\}$  a sufficiently rich collection of matrices so that all the  $A(k)$  can be written as in (4.4). For the sake of simplicity, we also denote  $\Delta(k) = [\Delta_1(k), \dots, \Delta_{n_\Delta}(k)]^T$  as the vector of uncertain parameters at times  $k$ .

As an example, consider a simple network with 3 nodes running a gossip consensus algorithm and let us look only at nodes 1 and 2, which we assume to have 3 different dynamics matrices

$$Q_{12} = \begin{bmatrix} 0.5 & 0.5 & 0 \\ 0.5 & 0.5 & 0 \\ 0 & 0 & 1 \end{bmatrix}, Q_{21} = \begin{bmatrix} 0.25 & 0.75 & 0 \\ 0.75 & 0.25 & 0 \\ 0 & 0 & 1 \end{bmatrix}, Q_{11} = Q_{22} = I$$

where  $Q_{11}$  and  $Q_{22}$  represent missed transmissions from node 1 and node 2 respectively. For that case, we can design the matrices  $A_0$  and  $A_\ell$  to be

$$A_0 = Q_{12}, A_1 = \begin{bmatrix} 0.5 & -0.5 & 0 \\ -0.5 & 0.5 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

and matrix  $Q_{11} = Q_{22} = A_0 + A_1$ ,  $Q_{12} = A_0$  and  $Q_{21} = A_0 - 0.5A_1$ . Therefore, for 3 possible transmission matrices we only require 1 uncertainty (i.e.,  $n_\Delta = 1$ ) and reduce the complexity of the algorithm.

The dynamics of the system can now be cast into an LPV model with uncertainty in the time-varying matrix  $A(k)$ . Indeed, the dynamics in (4.2) can be rewritten as:

$$x(k+1) = \left( A_0 + \sum_{\ell=1}^{n_\Delta} \Delta_\ell(k) A_\ell \right) x(k) + B(k)u(k). \quad (4.5)$$

Detecting a fault in a worst-case scenario amounts to finding whether there exists an admissible initial condition  $x(0)$  such that a given sequence of observations,  $y_k^i$ , can be generated by the dynamics in (4.5) with  $u(k) = 0$  for  $k \in \{0, 1, \dots, N\}$ . Therefore, the knowledge of the structure of  $B(k)$  is not needed for fault detection.

A fault-free (ideal) SVO for (4.3) is a dynamical system that produces a sequence of sets  $X(k), k \geq 0$  such that each  $X(k)$  is the smallest set that contains all possible values of the state  $x(k)$  of (4.3) that are compatible with the zero inputs  $u(0) = u(1) = \dots = u(k-1) = 0$  and the observed outputs  $y^i(0), y^i(1), \dots, y^i(k)$  of node  $i$ .

**Assumption 4.1** (bounded state). *For a “fault-free” system, the following holds:  $\forall k < N, \forall i : 1 \leq i \leq n_x, |x_i(k)| < c$  for a given constant  $c$ .*

□

Assumption 4.1 is sustained by the fact that a non-faulty gossip algorithm has a bounded state. Therefore, a node receiving a measurement indicating the absolute value of the state of a neighbor being larger than  $c$  can trivially detect the occurrence of the fault. Assumption 4.1 is fundamental for enclosing the initial state in a polytope and compute the set  $X(k)$  as described in the next proposition.

To prepare the proposition, we introduce some notation. A polytope at time  $k$  is defined as  $\text{Set}(M, m) := \{q : Mq + m \leq 0\}$  whereas we also introduce the notation  $M_{\Delta^\star}(k)$  and  $m_{\Delta^\star}(k)$  to refer the polytope for a particular instantiation  $\Delta^\star$  of the uncertainties. Similarly,  $A_{\Delta^\star}$  refers to the particular instantiation of the dynamics matrix using  $\Delta^\star$  value for the uncertainties.

We also recall the definition of the Fourier-Motzkin elimination method as

**Definition 4.2** (Fourier-Motzkin elimination method [Tel82]). *Take a polytope described by  $\left\{ \begin{bmatrix} x \\ y \end{bmatrix} \in \mathbb{R}^{n_x+n_y} : A \begin{bmatrix} x \\ y \end{bmatrix} \leq b \right\}$ . The Fourier-Motzkin elimination method is a function*

$$(A_{FM}, b_{FM}) = FM(A, b, n_x)$$

such that

$$A_{FM}y \leq b_{FM} \Leftrightarrow \exists_{x \in \mathbb{R}^{n_x}} : A \begin{bmatrix} x \\ y \end{bmatrix} \leq b.$$

Intuitively, we will compute the polytope containing the state for each of the vertices of the hypercube containing the vector  $\Delta(k)$ . For that reason, we show how to compute for a particular vertex (i.e., when  $\Delta(k)$  is a constant equal to one of the hypercube vertices) and then compute the convex hull of all the sets.

**Proposition 4.1** ( $X(k+1)$  computation [ST99]). *Consider a system described by (4.5), with  $u(\cdot) \equiv 0$ , and where  $x(k)$  denotes the corresponding state at times  $k$ , for  $k \geq 0$ . Further assume that*

- $x(0) \in X(0)$ , where  $X(0) := \text{Set}(M_0, m_0)$ , for some matrix  $M_0$  and vector  $m_0$  with appropriate dimensions;
- $\Delta(k) \equiv \Delta^\star$ , for some (constant) vector  $\Delta^\star$  and all  $k \geq 0$ ;
- $A_0 + A_{\Delta^\star}$  is non-singular.

Then, the set  $X(k+1) := \text{Set}(M_{\Delta^\star}(k+1), m_{\Delta^\star}(k+1))$ , which contains all the possible states of the system at time  $k+1$ , can be described at the expenses of the previous set-valued estimates ( $X(k) := \text{Set}(M(k), m(k))$ ) as the set of points,  $\mathbf{x}$ , satisfying the equation

$$\underbrace{\begin{bmatrix} M(k)(A_0 + A_{\Delta^\star})^{-1} \\ C_i(k+1) \\ -C_i(k+1) \end{bmatrix}}_{M_{\Delta^\star}(k+1)} \mathbf{x} \leq \underbrace{\begin{bmatrix} -m(k) \\ y^i(k+1) \\ -y^i(k+1) \end{bmatrix}}_{-m_{\Delta^\star}(k+1)} \quad (4.6)$$

where

$$A_{\Delta^\star} = \sum_{\ell=1}^{n_\Delta} \Delta_\ell^\star A_\ell$$

and  $\Delta_\ell^\star$  is the realization of the uncertainty for the current transmission time. When the dynamics matrices are not invertible, the set is given by solving the inequality relating the current time  $\mathbf{x}$  and the previous time with  $\mathbf{x}^-$

$$\begin{bmatrix} I & -A_0 - A_{\Delta^\star} \\ -I & A_0 + A_{\Delta^\star} \\ C_i(k+1) & 0 \\ -C_i(k+1) & 0 \\ 0 & M(k) \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{x}^- \end{bmatrix} \leq \begin{bmatrix} 0 \\ 0 \\ y^i(k+1) \\ -y^i(k+1) \\ -m(k) \end{bmatrix} \quad (4.7)$$

and applying the Fourier-Motzkin elimination method [KG87] (see Definition 4.2) to remove the dependence on  $\mathbf{x}^-$  and obtain the set described by  $M_{\Delta^\star}(k+1)\mathbf{x} \leq -m_{\Delta^\star}(k+1)$ .

Inequality (4.7) can be extended to a generic horizon obtaining the following:

$$\begin{bmatrix} I & -\tilde{A}_0^k & \cdots & 0 \\ -I & \tilde{A}_0^k & \cdots & 0 \\ I & 0 & \cdots & 0 \\ -I & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ I & 0 & \cdots & -\tilde{A}_{N-1}^k \\ -I & 0 & \cdots & \tilde{A}_{N-1}^k \\ C_i(k+1) & 0 & \cdots & 0 \\ -C_i(k+1) & 0 & \cdots & 0 \\ 0 & C_i(k) & \cdots & 0 \\ 0 & -C_i(k) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & C_i(k+1-N) \\ 0 & \cdots & 0 & -C_i(k+1-N) \\ 0 & M(k) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & M(k+1-N) \end{bmatrix} \begin{bmatrix} \mathbf{x}(k+1) \\ \vdots \\ \mathbf{x}(k+1-N) \end{bmatrix} \leq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ y^i(k+1) \\ -y^i(k+1) \\ y^i(k) \\ -y^i(k) \\ \vdots \\ y^i(k+1-N) \\ -y^i(k+1-N) \\ -m(k) \\ \vdots \\ -m(k+1-N) \end{bmatrix} \quad (4.8)$$

where the notation  $\mathbf{x}(k+1-N)$  is a variable to constrain the state at  $N$  time instants before the current time and  $\tilde{A}_n^k := (A_0 + A_{\Delta(k)}) \cdots (A_0 + A_{\Delta(k-n)})$ .

□

The previous proposition describes the set of possible states at time  $k+1$  for a particular instantiation of  $\Delta(k)$ , which considers no uncertainty in the system. As an example to illustrate the SVO computations, assume an abstract system described by the Linear Time-Invariant (LTI) model:

$$\begin{cases} x(k+1) = \begin{bmatrix} 0.75 & 0.25 \\ 0.25 & 0.75 \end{bmatrix} x(k) + 0.1d(k) \\ y(k) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(k) + v(k) \end{cases} \quad (4.9)$$

where  $\forall k \geq 0 : |v(k)| \leq 0.1$  and an initial state uncertainty  $\forall i \in \{1, 2\} : |x_i(0)| \leq 1$ . The system has invertible dynamics and the set  $X(1) = \text{Set}(M(1), m(1))$  given by

$$M(1) = \begin{bmatrix} 1.5 & -0.5 & -0.15 & 0.05 \\ -1.5 & 0.5 & 0.15 & -0.05 \\ -0.5 & 1.5 & 0.05 & -0.15 \\ 0.5 & -1.5 & -0.05 & 0.15 \\ 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}, m(1) = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0.1 \\ 0.1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

for the measurement  $y(1) = 0$ . The set  $X(1)$  is exact as we have assumed an LTI with no uncertainty in its dynamics and depends on the variables  $[\mathbf{x}^\top \mathbf{d}^\top]^\top$ . The set  $X(1)$  can be

described solely by the variable  $\mathbf{x}$  performing an elimination of the  $\mathbf{d}$  variable, obtaining

$$M(1) = \begin{bmatrix} 5 & -15 \\ -5 & 15 \\ 1 & 0 \\ -1 & 0 \end{bmatrix}, m(1) = \begin{bmatrix} -12 \\ -12 \\ 0.1 \\ 0.1 \end{bmatrix}$$

which we have depicted in Figure 4.1. In the case of an LTI, the methods described in the remainder of this section are not required since the exact set  $X(k)$  can be obtained.

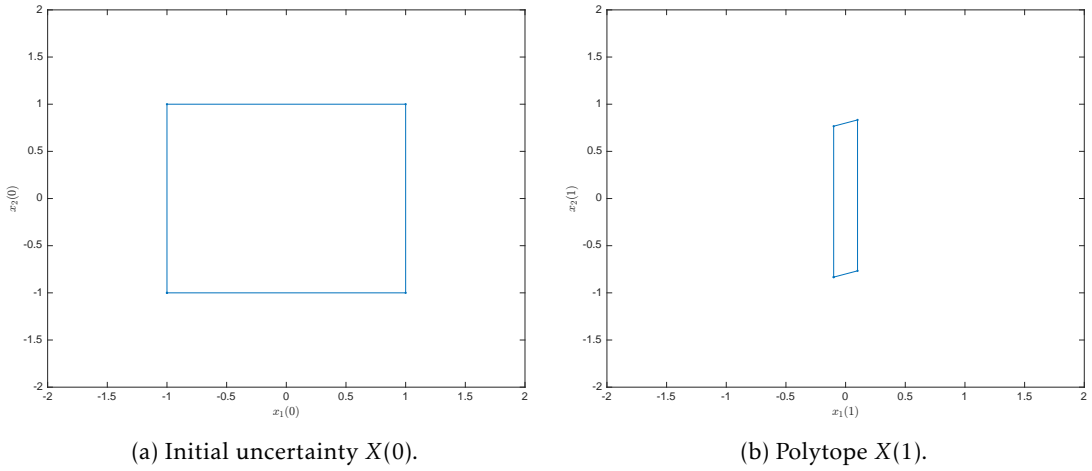


Figure 4.1: Example of the sets produced by the SVOs.

In order to compute the set  $X(k+1)$ , one would need to make the union of the sets for all possible instantiations of the uncertainties. As a consequence, set  $X(k+1)$  is, in general, non-convex which renders its calculation computationally heavy. For that reason, we are interested here in polytopical SVOs that produce the smallest sets of the form  $\tilde{X}(k) := \text{Set}(M(k), m(k))$  that contain the sets  $X(k)$  produced by the fault-free (ideal) SVO. Polytopical SVOs thus produce the smallest over-approximation of the sets produced by the ideal SVO.

For a given *horizon*,  $N$ , let the coordinates of each vertex of the hypercube  $\mathcal{H} := \{\delta \in \mathbb{R}^{n_\Delta N} : |\delta| \leq 1\}$  be denoted by  $\theta_i, i = 1, \dots, 2^{n_\Delta N}$ . Using (4.6) (or (4.7)), let us compute  $X_{\theta_i}(k)$ . Thus, the smallest set comprising all possible states of the system described by (6), with  $|\Delta_\ell(k)| \leq 1$  and  $u(\cdot) \equiv 0$ , at time  $k+1$  can be obtained by

$$\tilde{X}(k+1) = \text{co}\left(\bigcup_{\theta_i \in \mathcal{H}} \text{Set}(M_{\theta_i}(k+1), m_{\theta_i}(k+1))\right) \quad (4.10)$$

where  $\text{co}()$  denotes the convex hull. The vertices  $\theta_i$  should not be confused with the network agents, since they represent the possible combinations of the uncertainty parameters. The convex hull in (4.10) can be performed using the methods described in [RSSA10], [RS13]. It is straightforward to conclude that  $X(k+1) \subseteq \tilde{X}(k+1)$ . We recall Proposition 6.2 in [Ros11] for completeness, but considering a less restrictive condition where  $\gamma_N \leq 1$ .

**Proposition 4.2** (Growth of  $\tilde{X}(k)$ ). *Consider a system described by (4.5) with  $x(0) \in \tilde{X}(0)$  and  $u(k) = 0, \forall k$ , and suppose that there exists an  $N \geq 0$  such that*

$$\gamma_N := \max_{\substack{\Delta(k), \dots, \Delta(k+N) \\ |\Delta(m)| \leq 1, \forall m \\ k \geq 0}} \left\| \prod_{j=k}^{k+N} \mathcal{A}(j) \right\| \leq 1,$$

and where

$$\mathcal{A}(j) := \left[ A_0 + \sum_{\ell=1}^{n_\Delta} \Delta_\ell(j) A_\ell \right].$$

Then, it is possible to find a set  $X^o(k), \forall k$  with uniformly bounded hypervolume and number of vertices, such that  $\tilde{X}(k) \subseteq X^o(k)$ .

□

The proof follows the exact same steps as that of Proposition 6.2 in [Ros11]. The less restrictive condition is met for doubly stochastic matrices  $A(k)$  since the hyper-parallelepiped overbound for the initial state also includes the remaining set-valued estimates.

In summary, Proposition 4.2 states that the volume of  $\tilde{X}(k)$  is uniformly bounded for all  $k \geq 0$ , and that there is a hyper-parallelepiped that, at each time, contains the set  $\tilde{X}(k)$ , and has a uniformly bounded distance between any two vertices, for all  $k \geq 0$ .

Notice that the method provided before to compute  $M(k)$  and  $m(k)$  for the “fault-free” model, gives a set where the measurements can take values. Whenever this operation results in an empty set, the “fault-free” virtual system cannot generate the real system measurements and a fault is detected. In addition, in reference to Proposition 4.2, we can always derive a bounded set with a finite number of vertices to contain the set of actual possible states,  $\tilde{X}(k)$ .

The complexity of the algorithm to compute the set-valued estimates for the state is exponential in nature, since the number of vertices of the hypercube to be considered is  $2^{n_\Delta N}$ . The number of uncertainties, in a worst-case scenario, is equal to the number of vertices of the connectivity graph, as we can trivially associate each uncertainty with each possible communication link and define appropriate matrices  $A_\ell$  in (4.5).

In order to reduce the SVO complexity, it is essential to either consider a smaller horizon or decrease the number of edges in the connectivity graph relevant to our problem. One of the main contributions to be presented later in this chapter is the guarantee that, under mild assumptions, detection can be guaranteed for a sufficiently large number of observations. In practical cases, this amounts to setting the horizon  $N$  to a large value. However, the combinatorial behavior of the detection problem renders the computation of the SVO intractable, motivating the need to use smaller horizons. In other words, the horizon used by the algorithm may be small, so as to guarantee its practical implementability and still performing the detection at the expenses of a longer detection time.

In order to reduce the computational complexity, an alternative method consists of using hyper-parallelepiped overapproximations instead of computing the exact set by means of

executing the Fourier-Motzkin elimination method. For the purpose of fault detection and isolation, the main focus is on checking whether the observations are complying with the model and known bounds for the signals. With the objective of reducing the conservatism introduced by the hyper-parallelepiped, one can set the horizon to a large value, and in each time step, solve a linear problem that checks if there is a point in the set produced by the SVOs satisfying all its restrictions. Such a procedure is exact for all time instants up to the first approximation (i.e., before  $N$  iterations). After each overbounding of the set, which occurs every  $N$  time steps, conservatism is added due to the inclusion of states that are effectively incompatible with the observations and dynamics of the system. Nevertheless, having a large horizon reduces this conservatism. The gain in computational complexity comes from the fact that there exists efficient algorithms to solve linear programs in comparison with the doubly exponential complexity of the Fourier-Motzkin elimination method. For further details, the interested reader can check the implementation of the SVOs for fault detection in [CRS15].

We now introduce the concept of  $N_d^*$  as being the minimum horizon ensuring that the observer can get better estimates for nodes with maximum number of hops equal to  $d$ . Such definition is important to reduce the number of necessary edges by discarding irrelevant information in a worst-case perspective, when the horizon is smaller than the theoretical value of  $N_d^*$ .

**Definition 4.3** ( $N_d^*$ ). *Consider a node  $i$  running an SVO and any node  $q$  with a hop distance smaller than or equal to some generic value  $d$  to node  $i$ , i.e.,  $\text{dist}(i, q) \leq d$ . The quantity  $N_d^*$  is defined as the minimum horizon value for which there exists a sequence of transmissions such that*

$$P_q \tilde{X}(k + N_d^*) \subset P_q \tilde{X}(k), \forall_{q, k \geq 0}$$

where  $P_q$  is the projection operator on the  $q$ -th dimension.

Definition 4.3 formally introduced the concept of minimum horizon to estimate nodes with  $d$  hops of distance to the detector (not necessarily a finite set of points due to the convex hull operation). The value for  $N_d^*$  can be computed by constructing a sequence that sequentially introduces second degree neighbors interleaving with communications with direct neighbors, and then in a similar fashion for the remaining neighbors (this sequence is going to be formally introduced for the results of asymptotic accuracy later in this chapter). Thus, for  $d_1 \leq d_2$  we have  $N_{d_1}^* \leq N_{d_2}^*$  (i.e., the larger the number of hops between the detector and the farthest node, the larger the value of  $N_d^*$ ).

The next proposition places a bound on the number of edges and nodes to be considered if the horizon is smaller than or equal to  $N_d^*$ .

**Proposition 4.3** (SVO with local information). *Let a node  $i$  be running an SVO of a system described by (4.5) with  $x(0) \in \tilde{X}(0) := \{z \in \mathbb{R}^{n_x} : \|z\|_\infty \leq c\}$ , signal  $u(k) = 0, \forall k \geq 0$  and  $N \leq N_d^*$ .*

*Then, for any two nodes  $q_1$  and  $q_2$  with a hop distance to node  $i$  greater than  $d$  or equal to  $d$  but that share a neighbor with hop distance to  $i$  equal to  $d - 1$ , i.e.,*

- $\text{dist}(i, q_1) > d$ ;
- $\text{dist}(i, q_2) > d$ ;

or

- $\text{dist}(i, q_1) = d$ ;
- $\text{dist}(i, q_2) = d$ ;
- $\exists j : (q_1, j) \in E \wedge (q_2, j) \in E, \text{dist}(i, j) = d - 1$ ,

we get  $\forall k, P_{q_1} \tilde{X}(k) = [-c, c]$  and  $P_{q_2} \tilde{X}(k) = [-c, c]$ , where  $P_q$  is the projection operator on the  $q$ -th dimension and  $c$  is the constant in Assumption 4.1 and  $\tilde{X}(k)$  is the set generated by the SVO.

□

*Proof.* The case when  $\text{dist}(i, q_1) > d$  and  $\text{dist}(i, q_2) > d$  is a trivial consequence of the definition of  $N_d^\star$ . If  $N \leq N_d^\star$ , no possible sequence of transmissions exists of size  $N$  that allows node  $i$  to estimate  $q_1$  and  $q_2$ . Similarly, the result is also trivial for the case when  $N < N_d^\star$  as there is no sequence of size  $N$  to estimate nodes of hop distance greater than or equal to  $d$ .

We shall now prove the result when  $N = N_d^\star$ , the hop distance of  $q_1$  and  $q_2$  to  $i$  is  $d$  and the existence of a node  $j$  as in the statement of the theorem. Start by noticing that given the horizon  $N_d^\star$ , three situations can occur: i) there exists a sequence that allows node  $i$  to estimate the state of  $q_1$  and another sequence for node  $i$  to estimate  $q_2$ , both of length  $N_d^\star$ ; ii) there exists a sequence in the same conditions but only for one of the nodes; and, iii) there does not exist any sequence that allows to estimate either of those nodes.

The conclusion is straightforward for iii) since any sequence cannot make estimates of both  $q_1$  and  $q_2$ . In case i), the important step is to note that the sequence to determine the value of  $q_1$  is exactly of size  $N_d^\star$  and therefore cannot determine the value of  $q_2$  and the converse applies for the sequence to determine the value of  $q_2$ . Let us define  $\Delta^{q_1}$  as the instantiation of the uncertainties corresponding to the transmission to determine  $q_1$  and conversely  $\Delta^{q_2}$  for the sequence to determine  $q_2$ . The above translates into

$$P_{q_1} X_{\Delta^{q_1}}(k + N_d^\star) = [x_{q_1}(k + N_d^\star), x_{q_1}(k + N_d^\star)]$$

$$P_{q_2} X_{\Delta^{q_1}}(k + N_d^\star) = [-c, c]$$

or

$$P_{q_1} X_{\Delta^{q_2}}(k + N_d^\star) = [-c, c]$$

$$P_{q_2} X_{\Delta^{q_2}}(k + N_d^\star) = [x_{q_2}(k + N_d^\star), x_{q_2}(k + N_d^\star)]$$

where we recall that  $X_{\Delta^\star}(\cdot)$  is the ideal set-valued estimates without any approximation for the sequence of instantiations of the uncertainties given by  $\Delta^\star$ . However, given that node  $q_1$  and



$q_2$  have a common neighbor  $j$ , node  $i$  cannot infer if the actual sequence in the real system corresponds to  $\Delta^{q_1}$  or  $\Delta^{q_2}$ . Thus, regardless of the actual sequence of transmissions, the convex hull operation in (4.10) to obtain  $\tilde{X}(\cdot)$  means that

$$P_{q_1} \tilde{X}(k + N_d^*) = [-c, c]$$

and

$$P_{q_2} \tilde{X}(k + N_d^*) = [-c, c].$$

Given that for any  $N < N_d^*$  the conclusion of the proposition holds, from applying iteratively the above relationship, we get that  $\forall k, P_{q_1} \tilde{X}(k) = [-c, c]$  and  $P_{q_2} \tilde{X}(k) = [-c, c]$ . For the case ii), the same reasoning can be applied with the difference that either

$$P_{q_1} X_{\Delta_1}(k + N_d^*) = [-c, c]$$

or

$$P_{q_2} X_{\Delta_2}(k + N_d^*) = [-c, c]$$

and the same conclusion follows.  $\square$

The key reason for the result in Proposition 4.3 is that, to avoid dealing with the non-convex set  $X(k)$ , we resorted to the convex hull  $\tilde{X}(k)$ . In Proposition 4.3, we limited the analysis to pairs of nodes with  $d$  hop distance. However, the same issue arises for any number of neighbors of a node  $j$  with  $d - 1$  hop distance. If the horizon value is not sufficient to determine the value of all nodes, then the convex hull operation results in the same problem.

A corollary from Proposition 4.3 is that all neighbors satisfying the conditions can actually be removed from the estimation as the set-valued estimates for their state remains constant. To maintain the same model for the remaining nodes, a single node can be added that works as a perturbation. In doing so, selecting a horizon limits the modeled network to a local view from the perspective of the estimator. A direct practical consequence is that, given the need to consider small horizon values to save computational resources, the computational complexity is bounded by the local neighborhood of the estimator and dependent on node degree instead of the full size of the network.

A detection mechanism is only interesting in practice if its complexity scales well with the number of nodes in the network. We showed that the set can be computed using only local information without loss of accuracy if  $N \leq N_d^*$ . To produce accurate estimates, intuitively, we need all the information regarding observations that are available to build a smaller set at the expenses of propagating those observations with the system dynamics. However, we can relax this definition and discard old information that does not enhance the set-valued state estimate, according to the next theorem. We introduce the notation  $\tilde{X}^N(k) := \text{Set}(M^N(k), m^N(k))$  to explicitly state the set-valued estimates computed using the horizon value  $N$ .

## Chapter 4: Set-Valued Estimators

**Theorem 4.1.** Take a system as defined in (4.5) and consider an SVO, running in node  $i$ , with local information. If it is possible to find  $N$  such that

$$\forall n > N, \forall q, \exists n^* \leq N : P_q(\tilde{X}^{n^*}(k)) \subseteq P_q(\tilde{X}^n(k)),$$

then,  $\tilde{X}^N(k+1) \subseteq \tilde{X}^n(k+1)$ .

□

*Proof.* For a horizon  $N = 1$ , from equation (4.5), the set  $\tilde{X}^1(1)$  is obtained using  $\theta_i = 1, \dots, 2^{n_\Delta}$ . If a communication with node  $i$  happens then  $\theta_i = \theta_i^*$ , where  $\theta_i^*$  corresponds to an instantiation of the uncertainties for that communication. For a generic  $N$ , if the node did not communicate with any of its neighbors, then  $\tilde{X}^N(k)$  is computed using  $\theta_i \times \dots \times \theta_i$ , where  $\times$  represents the Cartesian product and is taken  $N$  times. A measurement is equivalent to setting  $\theta_i = \theta_i^*$  for a particular instant. From this fact, with the last observation measured at time  $k_q$  results in  $\forall n > k - k_q, P_q \tilde{X}^{k-k_q}(k) \subseteq P_q \tilde{X}^n(k)$ . By definition the set-valued estimates can only be improved with a higher horizon so we also have  $P_q \tilde{X}^n(k) \subseteq P_q \tilde{X}^{k-k_q}(k)$ . Thus,

$$P_q \tilde{X}^n(k) = P_q \tilde{X}^{k-k_q}(k). \quad (4.11)$$

Equation (4.11) simply states that we cannot improve the estimates for a node  $q$  by considering measurements older than the last communication with that node. The condition  $P_q \tilde{X}^{n^*}(k) \subseteq P_q \tilde{X}^n(k)$  means that the selected  $N$  is such that  $\forall q : (i, q) \in E$ , it exists  $\exists k_q : k - k_q \leq N, A(k - k_q) = Q_{iq}$ . Therefore, combining with (4.11), we cannot improve the estimates by considering any  $n > N$ . Thus, we reach the conclusion

$$\tilde{X}^N(k+1) \subseteq \tilde{X}^n(k+1).$$

□

The intuition behind Theorem 4.1 is that we do not need to consider past time instances prior to the last communication that we established with each node. The horizon value  $N$  must be sufficiently large as to have node  $i$  (the node running the SVO) communicating with all its neighbors and the previous time instants can be neglected.

**Remark 4.1** (Bound in the Horizon). From Theorem 4.1, the set  $\tilde{X}^N(k)$ , when  $N$  is selected such that there exists a transmission between all the neighbors and the node, and the modeled network is composed of local information only (neighbors with second-degree neighbors as perturbations), is the smallest possible set.

## 4.5 Fault Detection using Stochastic Set-Valued Observers (SSVO)

SVOs are deterministic and discard the probabilistic information of each event. They consider as admissible all states that can be generated by the considered LPV dynamics, regardless of

how likely they are. By taking into account the stochastic information in the definition of the SVO, one may decide to declare a fault when the observations are, in principle, possible, but have an exceedingly small probability of occurrence. This typically permits the earlier detection of attacks, at the expense of generating *false alarms*. The algorithm proposed in the sequel allows for controlling the probability of false alarms.

To better understand how probabilistic information can help detect faults, consider the 5-node complete network ( $n_x = 5$ ) and time horizon to detect the fault  $N = 20$ . Each node  $i$  takes a measurement  $x_i(0)$  of a quantity of interest and then starts a linear randomized gossip algorithm. Let us assume that the packet drop probability is known. In particular, let  $p_{\text{drop}} = 0.01$  where a packet drop is represented as a transmission from node  $i$  to itself, using the transmission matrix  $Q_{ii} = I$ . Each node is chosen with probability  $\frac{1}{n_x}$  and each matrix  $Q_{ij}$  representing a successful transmission from node  $i$  to  $j$  has probability  $\frac{w_{ij}}{n_x}$ .

If a node is not involved in a communication, it is only able to determine its own state. Suppose that the states of the agents start dissimilar from each other but that during the first  $N$  time steps, all agents are faulty and keep their states unchanged, i.e.,  $x(k) = x(0), \forall k \leq N$ . This fault is undetectable according to Definition 4.1, since there is a sequence of matrices  $A(k)$  that mimic the same behavior, which is a sequence of 20 failed transmissions due to the physical medium. Consequently, if the algorithm in the previous section is used,  $x(k) = x(0)$  must remain in the set  $\tilde{X}(k), \forall k$  and therefore a fault will not be detected. However, the probability of obtaining the sequence  $x(k) = x(0), \forall k \leq N$  is extremely small:

$$\text{Prob}\{x(k) = x(0), \forall 0 \leq k \leq 20\} = 10^{-40}$$

and is more likely to be a fault. The inability of the SVO to incorporate the probability associated with each event is, therefore, a significant drawback. Such an example motivates the introduction of Stochastic Set-Valued Observers (SSVOs) where the polytope containing the possible state is associated with a probability. The objective of this section concerns with extending the SVO concept to cope with the probability of getting a given sequence of measurements. With that target in mind, we introduce the definition of  $\alpha$ -confidence sets.

**Definition 4.4** ( $\alpha$ -confidence sets). *The set  $\tilde{X}(k)$  is an  $\alpha$ -confidence set at time  $k$  for a system of the form (4.3) with state  $x(k)$  if*

$$\text{Prob}[x(k) \in \tilde{X}(k)] \geq 1 - \alpha.$$

Consider the algorithm described in the previous subsection to generate the sets  $\tilde{X}(k)$  and recall that it included all matrices  $Q_{ij}$  by selecting a sufficiently rich collection of matrices  $A_\ell$ . The objective of this section is to construct the  $\alpha$ -confident set, as in Definition 4.4 as to associate the probability of the events in the fault detection. In essence, the collection of matrices  $A_\ell$  must be associated with a given confidence level  $\alpha$ .

Take the map  $\psi : \theta_i \mapsto E$  which gives the correspondence between the vertices of the hypercube  $\mathcal{H}$  and the edges in set  $E$  and let us collect the minimum number of vertices  $\theta_{ij}$  in  $\Theta$

such that  $\sum_{\theta_{ij}} w_{\psi(\theta_{ij})} \geq 1 - \alpha$ . The set for the SSVO  $\bar{X}(k)$  is then an  $\alpha$ -confidence set defined as:

$$\bar{X}(k) := \text{co}\left(\bigcup_{\theta_{ij} \in \Theta} \text{Set}(M_{\theta_{ij}}(k), m_{\theta_{ij}}(k))\right) \quad (4.12)$$

Computationally, it requires to sort the vertices  $\theta_{ij}$  according to probabilities  $w_{\psi(\theta_{ij})}$  as to construct  $\Theta$  and then determining  $M_{\theta_{ij}}(k)$  and  $m_{\theta_{ij}}(k)$  as before.  $\theta_{ij}$  depends on the selected edges and there can be multiple sets  $\Theta$  generating an  $\alpha$ -confidence set, with similar characteristics.

In the next Property, we establish that the set generated by the SSVO is an  $\alpha$ -confidence set. In this context, the parameter  $\alpha$  can be viewed both as the probability of false positives and also as a similar concept as the confidence interval for stochastic variables.

**Property 1.** *Take the definition of  $\bar{X}(k)$  as in (4.12). Then,  $\forall k, \bar{X}(k)$  is an  $\alpha$ -confidence set.*

*Proof.* The result is straightforward from the fact

$$\begin{aligned} \text{Prob}\left[x(k) \in \bigcup_{\theta_i \in \Theta} \text{Set}(M_{\theta_i}(k), m_{\theta_i}(k))\right] &\geq \sum_{\theta_i \in \Theta} w_{\psi(\theta_i)} \\ &\geq 1 - \alpha \end{aligned}$$

□

Property 1 establishes the SSVOs as a generalization of the SVOs since the set  $\bar{X}(k)$  is an  $\alpha$ -confidence set with  $\alpha = 0$  and, therefore, we have  $\bar{X}(k) \subseteq \tilde{X}(k)$ .

Taking advantage of the definition of SSVOs, we introduce Algorithm 1 for probabilistic detection of faults. The construction of the set  $\bar{X}(k)$  ensures that, with probability  $1 - \alpha$ , the state  $x(k)$  belongs to  $\bar{X}(k)$  and thus is an  $\alpha$ -confidence set.

---

**Algorithm 1** Detection using SSVO

---

**Require:** Set  $\bar{X}(0)$ , the probability matrix  $W$  and the confidence level  $\alpha$ .

**Ensure:** Computation at each time instant  $k$  of  $\bar{X}(k) : \text{Prob}[x(k) \in \bar{X}(k)] \geq \alpha$  and Fault Detection.

```

1: for each  $k$  do
2:   /* Finding the set  $\Theta$  */
3:    $\Theta = \arg \min \text{card}(\{\theta_{ij}\})$ 
4:   s.t.  $\sum w_{\psi(\theta_{ij})} \geq 1 - \alpha$ 
5:   /* Build the set  $\bar{X}(k+1)$  */
6:    $\text{SSVO\_iteration}(\Theta, \bar{X}(k), y(k+1))$ 
7:   /* Check if  $\bar{X}(k+1)$  is empty */
8:   if  $\bar{X}(k+1) = \emptyset$  then
9:     return System is faulty
10:  end if
11: end for

```

---

Notice that, in Algorithm 1, the function *SSVO\_iteration* is implementing the procedure to compute the set-valued estimates defined in (4.6) or (4.7), using the uncertainty values stored in  $\Theta$ . In essence, the SSVO propagation is exactly the same as the standard SVO except for the fact

that less uncertainties are considered in the hypercube, due to the fact that the vertices having low probability of occurring are not included. Detection is ensured if we make the bounded assumption as in Assumption 4.1, and also that the transmission selection procedure operates as described in Section 4.3. Detection guarantees will be provided later in this chapter, with a further discussion on the meaning of a detection using Algorithm 1.

## 4.6 Byzantine Consensus Algorithm

In this section, we describe how the information used to construct the set of possible states can be used to introduce a novel algorithm to compute consensus of intervals in a distributed way, and detect if a fault has occurred.

In a consensus system, we are referring to the agents running a distributed iterative algorithm that guarantees convergence of the state to its initial average value, i.e., aiming to satisfy (2.1). This problem can be tackled by a standard algorithm (such as [BGPS06]) and then, an SVO-based overlay to detect faults such as in [SRC<sup>+</sup>13]. In this section, an algorithm is introduced that incorporates the information used to construct the local estimate (i.e., a given node's estimate) of possible states and reduce conservatism by intersecting it with the state estimates from its neighbors. In the process, the set of possible states is reduced and the consensus solution is reached in finite time.

Each node runs an SVO to determine the set of possible states of all the nodes in the network. With a slight abuse of notation, we will denote  $\tilde{X}_i(k)$  for the set computed by node  $i$  which contains estimates for the states of all the nodes in the network using the measurements performed by node  $i$ . In general, the result of the Fourier-Motzkin elimination method produces a polytopic set with a bounded number of vertices. However, transmitting the set  $\tilde{X}_i(k)$  would mean communicating the matrix  $M_i(k)$  and vector  $m_i(k)$ , which define the set-valued state estimate  $\tilde{X}_i(k)$ . Since the dimension of  $M_i(k)$  depends on the number of vertices, we might need to communicate a large amount of information, which may not be feasible in many applications.

For that reason, we can overbound this set-valued estimates by a hyper-parallelepiped  $\text{Set}(\hat{M}_i(k), z_i(k))$ , with

$$\hat{M}_i(k) = I \otimes \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

and  $z_i(k) \in \mathbb{R}^{2n_x}$ , where  $z_i(k)$  is defined such that  $\text{Set}(\hat{M}_i(k), z_i(k)) := \{q : \hat{M}_i(k)q + z_i(k) \leq 0\}$  contains  $\tilde{X}_i(k)$ . Using this approach,  $z_i(k)$  will be the only vector that we need to transmit between neighbors. Thus, the  $z_i(k)$ 's represent state boundaries for the other agents and are obtained through the previously described algorithm to compute the SVO (4.6) or (4.7), by using the local information available when communicating with the neighbors.

An important issue here is the possible large conservativeness of the upperbounding of  $\tilde{X}_i(k)$  by a hyper-parallelepiped set. However, in order to minimize this issue, one can increase the horizon and consider more measurements in building  $\tilde{X}_i(k)$  and, therefore, getting better

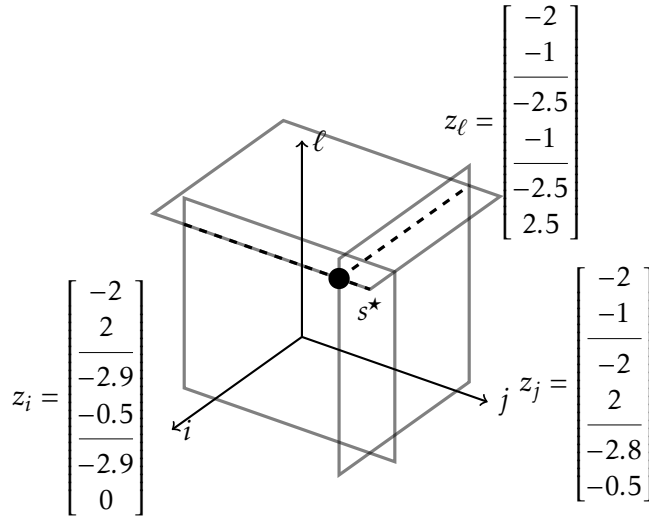


Figure 4.2: Example of the set-valued estimates boundaries of node  $i$  (yellow), node  $j$  (green) and node  $\ell$  (red), where for each node there is no uncertainty regarding its own state and where  $s^*$  represents the full state of the system that is contained in all three state boundaries.

estimates [RS13]. Thus, there is a trade-off between speed of computation of the SVO and its conservativeness when selecting the horizon.

The algorithm (see flow chart in Figure 4.3) can be briefly described as follows: in each discrete time instant, each node that does not communicate with its neighbors updates its set-valued state estimates of the corresponding SVO using (4.6) or (4.7). If node  $i$  communicates with node  $j$ , then it proceeds to an intersection of both set-valued state estimates motivated by the fact that  $z_i$  and  $z_j$  are estimates for the state boundaries of all nodes constructed using the information available to node  $i$  and  $j$ , respectively. The intersection step is described using the maximum function ( $z$  variables represent intervals and were defined to have the minimum and the maximum multiplied by  $-1$ , see Figure 4.2 for a numeric example) by operating on the state of the two communicating nodes  $i$  and  $j$

$$z_i(k) = z_j(k) = \max(z_i(k), z_j(k)) \quad (4.13)$$

where the max function, which operates row-wise, returns a column vector of the same length.

The result of performing the intersections can be described by

$$s^* = \left[ [z_1]_1^T, [z_1]_2^T, \dots, [z_{n_x}]_{2n_x-1}^T, [z_{n_x}]_{2n_x}^T \right]^T$$

and represents the collaborative estimation performed by all the nodes since  $s^* \in \text{Set}(\hat{M}_i, z_i)$  and  $s^* \in \text{Set}(\hat{M}_j, z_j)$ . The concept of  $s^*$  and the state boundaries generated by each node with the corresponding  $z$  variable is illustrated in Figure 4.2. A fault is declared by node  $i$ , whenever it receives  $z_j$  from node  $j$ , with  $[z_i]_{2j-1} > [z_j]_{2j-1} \vee [z_i]_{2j} > [z_j]_{2j}$ . This means that their estimates do not intersect and there is no vector  $s^*$  of possible states that satisfies the observations made by the different nodes in the network.

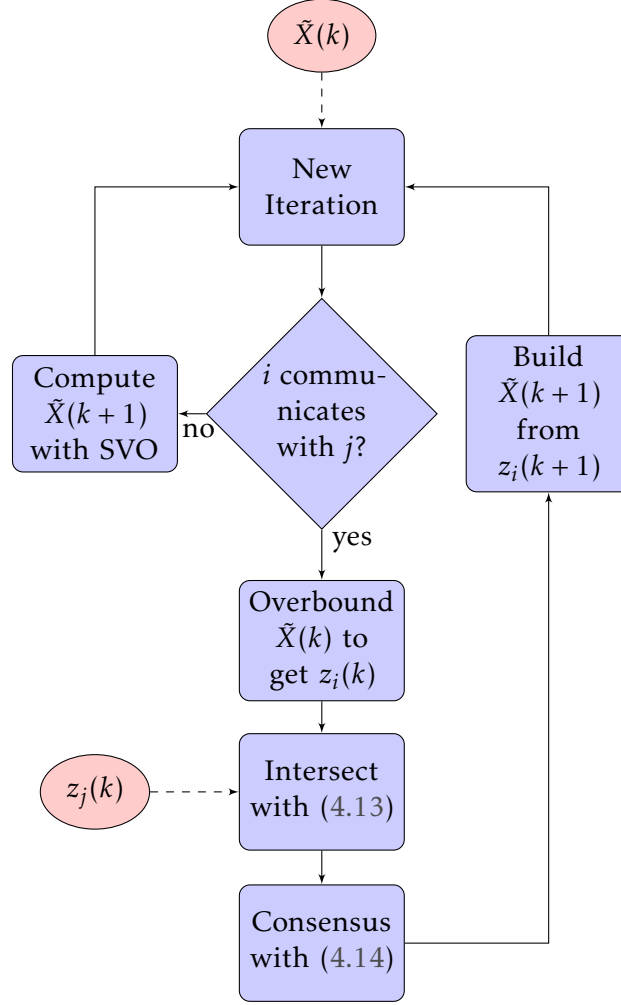


Figure 4.3: Flowchart of the algorithm with the intersection phase to share observations between neighbors.

At each time  $k$ , the consensus phase runs in both communicating nodes and is defined for node  $i$  communicating with node  $j$  by the following linear iteration, similarly to what is done in [BGPS06]:

$$z_i(k+1) = \left[ \left( \frac{1}{2}(e_i - e_j)(e_j - e_i)^\top + I_{n_x} \right) \otimes I_2 \right] z_i(k) \quad (4.14)$$

where, as previously mentioned, the variable  $z_i$  is the vector-valued estimate of node  $i$  of all the states of the nodes of the network. It should be noticed that, for node  $i$ , we may have  $[z_i]_{2i} \neq [z_i]_{2i-1}$  if there is uncertainty associated to it.

As a remark, the algorithm defined through (4.13) and (4.14), and in Figure 4.3, not only computes the consensus value of its state, but also keeps estimates for all the remaining ones, using observations made by the node itself and its neighbors. This algorithm differs from the one proposed in [SRC<sup>+</sup>13] in the sense that the estimates of the SVO in each node are used to compute the state boundaries  $z_i(k)$  at each time instant and then shared with the neighbors when communicating, producing an intersection of measurements that is then subjected to the

standard gossip consensus step.

**Definition 4.5.** We say that a linear distributed algorithm taking the form of (4.1):

(i) converges almost surely to average consensus if

$$\lim_{k \rightarrow \infty} x_i(k) = x_{av} := \frac{1}{n_x} \sum_{i=1}^{n_x} x_i(0) , \quad \forall_{i \in \{1, \dots, n_x\}}$$

almost surely.

(ii) converges in expectation to average consensus if

$$\lim_{k \rightarrow \infty} \mathbb{E}[x_i(k)] = x_{av} , \quad \forall_{i \in \{1, \dots, n_x\}}.$$

(iii) converges in second moment to average consensus if

$$\lim_{k \rightarrow \infty} \mathbb{E}[(x_i(k) - x_{av})^2] \rightarrow 0 , \quad \forall_{i \in \{1, \dots, n_x\}}.$$

Where  $\mathbb{E}$  is the expected value operator. The next theorem proves asymptotic convergence as in Definition 4.5 and we delay the presentation of its finite-time property as a main result of this chapter. Notice that, we are looking at the evolution of the estimates of each SVO as a consequence of the algorithm. For that reason, we must consider the fault-free model for the algorithm that merges the received estimates even though the actual state might be corrupted since it is guaranteed the state is within the estimates as otherwise a fault would have been detected.

**Theorem 4.2.** Take the SVO-based consensus algorithm defined in this section. If the support graph of the matrix of probabilities  $W$  is strongly connected, then the algorithm converges in:

- expectation
- mean square sense
- almost surely.

□

*Proof.* The proof follows a similar reasoning as in [BGPS06]. We start by stacking each node own estimates  $[z_i]_{2i-1}$  and  $[z_i]_{2i}$  and prove the convergence of the whole system. Let us introduce variable  $\mathbf{z}$ :

$$\mathbf{z} = \begin{bmatrix} [z_1]_1 \\ [z_1]_2 \\ [z_2]_3 \\ [z_2]_4 \\ \vdots \\ [z_{n_x}]_{2n_x-1} \\ [z_{n_x}]_{2n_x} \end{bmatrix}$$



with  $\mathbf{z} \in \mathbb{R}^{2n_x}$ , where  $n_x$  is the number of nodes. Then, one can write

$$\mathbf{z}(k+1) = U_k \mathbf{z}(k)$$

where  $U_k$  is a matrix randomly selected from  $\{Q_{ij}\}$ , where the matrices  $Q_{ij}$  respect the given structure if we consider that each node has two states, given by

$$Q_{ij} = \left( \frac{1}{2}(e_i - e_j)(e_j - e_i)^\top + I_{n_x} \right) \otimes I_2$$

for each pair of nodes  $i$  and  $j$  communicating with each other with probability  $w_{ij}$  gathered in the probability matrix  $W$ .

We start by proving convergence in expectation since convergence in mean square will be derived from this result. Let us define

$$R = \mathbb{E}[U_k].$$

Then,

$$\mathbb{E}[\mathbf{z}(k+1)] = R \mathbb{E}[\mathbf{z}(k)]$$

due to the probability distributions  $w_{ij}$  being independent. By applying iteratively, we get

$$\mathbb{E}[\mathbf{z}(k+1)] = R^k \mathbb{E}[\mathbf{z}(0)].$$

Rearranging the variables using the transformation  $T^\top Q_{ij} T$  with

$$[T]_{ij} = \begin{cases} 1, & \text{if } j = 2i - 1 \wedge i \leq n_x \\ 1, & \text{if } j = 2(i - n_x) \wedge i > n_x \\ 0, & \text{otherwise} \end{cases}$$

we get

$$T^\top R T = I_2 \otimes \left( \left(1 - \frac{1}{n_x}\right) I_{n_x} + \frac{1}{n_x} W \right).$$

The eigenvalues of  $R$  are the eigenvalues of  $(1 - \frac{1}{n_x})I_{n_x} + \frac{1}{n_x}W$  counted twice. We can use the fact that

$$\lambda\left(\left(1 - \frac{1}{n_x}\right)I_{n_x} + \frac{1}{n_x}W\right) = \left(1 - \frac{1}{n_x}\right) + \frac{1}{n_x}\lambda(W)$$

and since  $W$  is a doubly stochastic matrix with a strongly connected support graph with all but one eigenvalues less than 1. The  $\lambda(W) = 1$  is associated to the eigenvector  $\mathbf{1}_{n_x}$ . Thus,  $\lim_{k \rightarrow \infty} R^k = I_2 \otimes \mathbf{1}_{n_x}/n_x$  which proves the convergence in expectation with rate equal to  $(1 - \frac{1}{n_x}) + \frac{1}{n_x}\lambda_2(W)$ , where  $\lambda_2$  is the second largest eigenvalue.

In order to prove convergence in the mean square sense, let us compute

$$\mathbb{E}[\mathbf{z}(k+1)^\top \mathbf{z}(k+1)] = R_2 \mathbb{E}[\mathbf{z}(k)^\top \mathbf{z}(k)]$$

where  $R_2 = R$  due to the fact that  $Q_{ij}^\top Q_{ij} = Q_{ij}$ . Therefore, using the same argument as for the convergence in expectation, the algorithm converges in the mean square sense with the same

rate as the convergence in expectation. Almost surely convergence is given by using the fact that  $\mathbb{E}[z(k+1)] = R^k \mathbb{E}[z(0)]$ , which means that convergence is achieved at an exponential rate. Using the Borel-Cantelli first lemma [Bor09, Can17], the sequence converges almost surely.  $\square$

The previous theorem shows the asymptotic convergence of the algorithm and found a closed-form for its convergence rates. The result is useful when characterizing its behavior in the presence of approximations, since we overbounded the set  $\tilde{X}_i(k)$  with a hyper-parallelepiped to reduce the amount of information that is communicated at each time instant. However, such a result only considers each node current state which is known and does not need to be estimated since it is measured every time instant. We defer to the next section a result of finite-time convergence that provides a faster convergence by exploring the SVO estimates and intersection during each communication.

## 4.7 Theoretical overbound on the fault signal

An important issue regarding any fault detection method is the “maximum impact” of a fault in the system. The meaning of “maximum impact” depends on the specific application. Whereas in a physical system it makes sense to measure the energy of the fault signal being injected, in the case of consensus the maximum impact is given by the sum of the fault signal at each time instant. More generally, we can consider any function  $f(u_k)$ , where  $u_k$  stacks all the values of signal  $u$  until time instant  $k$ .

For the case of a physical system, function  $f$  takes the form

$$\frac{1}{N} \sum_{k=0}^N \|u(k)\|^2$$

whereas for the consensus case,  $f$  is a linear combination of fault signal  $u$  of the form

$$\frac{1}{N} \sum_{k=0}^N u(k).$$

As an example, consider a 3-node network where all the nodes can communicate among them. Now take two fault signals for two time slots  $u_1 = \mathbf{1}_2$  and  $u_2 = 10^6 \begin{bmatrix} 1 \\ -1 \end{bmatrix}$ . Signal  $u_1$  has an energy equal to 1 while  $u_2$  has  $10^{12}$ . Using the energy of the signal as a metric,  $u_2$  should have a higher impact on the system, even though its real impact on the final consensus value is zero, while for the signal  $u_1$  shifts the true steady state in  $2/3$ .

A theoretical bound can be computed *a priori* using the SVO framework for the maximum impact of a fault. We start by looking at the worst possible attack that is not guaranteed to be detected. Let us borrow the definitions from [RS11]:

$$(A_N, b_N) = FM \left( \begin{bmatrix} M_N \\ -M_N \\ \tilde{M}_0 \\ \tilde{M}_W \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ \tilde{m}_0 \\ \tilde{m}_W \end{bmatrix}, 2n_x \right) \quad (4.15)$$

where the *FM* stands for the Fourier-Motzkin elimination method [KG87] and:

$$\tilde{M}_0 = \begin{bmatrix} \text{diag}(M_0, M_0) & 0 \end{bmatrix}, \tilde{m}_0 = \begin{bmatrix} m_0 \\ m_0 \end{bmatrix},$$

$$\tilde{M}_W = \begin{bmatrix} 0 & \text{diag}(M_d, \dots, M_d) \end{bmatrix},$$

$$\tilde{m}_W = \begin{bmatrix} m_d^\top & \dots & m_d^\top \end{bmatrix}^\top,$$

$$M_N = \left[ \begin{array}{cc|c} C_A & -C_B & \\ C_A A_A & -C_B A_B & \\ \vdots & \vdots & \\ C_A A_A^N & -C_B A_B^N & \end{array} \right] \bar{R},$$

$$\bar{R} = \begin{bmatrix} 0 & 0 & \dots & 0 \\ R_1^1 & 0 & \dots & 0 \\ R_1^2 & R_2^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ R_1^N & R_2^N & \dots & R_N^N \end{bmatrix},$$

$$R_i^k = \begin{bmatrix} C_A A_A^{k-i} B_A & -C_B A_B^{k-i} B_B \end{bmatrix}.$$

where  $M_d$  and  $m_d$  define the set of allowable realizations of  $u$ , i.e.,  $M_d$  and  $m_d$  are defined such that  $u(k) \in \text{Set}(M_d, m_d)$ ; and  $A_i$ ,  $B_i$ , and  $C_i$ , with  $i \in \{A, B\}$  are the matrices of the dynamics of two linear time-varying systems, as defined in (4.3) and further described in the sequel. With a slight abuse of notation, we write the product of  $N$  matrices  $A(k)$  as  $A^N = A(k)A(k-1)\dots A(k-N+1)$  for shorter notation.

The aforementioned definitions characterize the set of admissible inputs that make both models have the same outputs. In the following proposition, a theoretical threshold  $\gamma_{\min}$  for any function  $f$  of the input fault  $u$  is given. The value of  $\gamma_{\min}$  defines the maximum impact of a fault that is not guaranteed to be detected.

**Proposition 4.4** (Attacker signal bound). *Let us consider a “fault-free” system:*

$$S_A = \begin{cases} x_A(k+1) = A(k)x_A(k) \\ y_A(k) = C(k)x_A(k) \end{cases}$$

*and a faulty system:*

$$S_B = \begin{cases} x_B(k+1) = A(k)x_B(k) + B(k)u(k) \\ y_B(k) = C(k)x_B(k) \end{cases}$$

where  $u \in \mathbb{R}^{n_u}$ ,  $x_i \in \mathbb{R}^{n_x}$ ,  $y_i \in \mathbb{R}^2$ , initialized with the same initial conditions. Compute the pair  $(A_N, b_N)$ , which is the set for all possible values of  $u$  of the last  $N$  time instants, defined as in (4.15).

## Chapter 4: Set-Valued Estimators

---

Consider  $\gamma_{min}$  to be the theoretical threshold for the fault given as the result of the convex optimization

$$\gamma_{min} := \max_{A_N \xi \leq b_N} f(\xi),$$

where the vector  $\xi$  is a variable stacking all possible attacker signals  $u$  in the last  $N$  time instants and  $f$  be a generic function depending only on  $\xi$ . The fault is guaranteed to be detected if

$$f(u_k) > \gamma_{min}. \quad (4.16)$$

□

The result presented in Proposition 4.4 is a direct consequence of the definition of the set  $\{\xi : A_N \xi \leq b_N\}$ . The advantage of the representation in (4.16) is that the distinguishability problem is cast as an optimization or feasibility problem subject to linear constraints. Definition 4.1 has a clear connection to Proposition 4.4. The value of  $\gamma_{min}$  identifies detectable faults since any fault signal that is detectable will have an evaluation of function  $f$  higher than the theoretical  $\gamma_{min}$ . Proposition 4.4 was discussed in [SRC<sup>+</sup>13] for the quadratic norm function.

In the context of the two particular cases that we were describing, the threshold for the energy of the fault signal can be computed using

$$\gamma_{min} := \max_{A_N \xi \leq b_N} \xi^\top P \xi$$

with

$$P = \frac{1}{N} \text{diag}(0_{n_u}, I_{n_u}, \dots, 0_{n_u}, I_{n_u})$$

and the maximum impact for the consensus case is given by

$$\gamma_{min} := \max_{A_N \xi \leq b_N} P_c \xi. \quad (4.17)$$

with

$$P_c = \frac{1}{N} [0_{n_u}^\top, \mathbf{1}_{n_u}^\top, \dots, 0_{n_u}^\top, \mathbf{1}_{n_u}^\top].$$

In the case of consensus, if we define the true consensus value as  $x_{\text{true}}$ , using Proposition 4.4 with function (4.17) we get:

$$\mathbf{1}_{n_x} x(k+N) - x_{\text{true}} = \frac{\gamma_{min}}{n_x}$$

The value of  $\gamma_{min}$  decreases as  $N$  increases, as more information is considered and, therefore, the longer the sequence of observations, the smaller impact an attacker can have on the final consensus value while avoiding detection. Thus, increasing the observation horizon decreases the impact of undetectable faults on the final consensus value. Since the algorithm introduced in Section 4.6 produces better estimates than the distributed individual detection using an SVO per node, it ensures a smaller effect of undetectable faults.

Proposition 4.4 defines a possible categorization of the undetectable faults using their impact on the final value of consensus. Nevertheless, calculating  $\gamma_{min}$  *a priori* to determine what

value of  $N$  we should choose in order to meet a certain level of quality in the final consensus value, requires a combinatorial calculation. We recall that computing the set  $\text{Set}(A_N, b_N)$  is combinatorial both in the number of uncertainties and also in the horizon  $N$ . As an alternative, one can simply compute the set-valued estimates and at each time compute an overbound for  $\gamma_{\min}$  as the summation of all the edges of the polytopic set. If no fault was detected, the maximum change in the states is given by the difference between the maximum of the estimate interval and its minimum.

Parameter  $\gamma_{\min}$  is the smallest input before systems  $S_1$  and  $S_2$  are distinguishable in the sense that the measured output of the faulty system cannot be generated by the dynamics of the non-faulty one. As a consequence, we can use the same line-of-thought to derive the following result.

**Corollary 4.1** (Attacker signal bound for SSV0). *Consider a non-faulty system  $S_1$  and a faulty system  $S_2$  as in Proposition 4.4. Then, a fault is detectable in  $N$  measurements with a false alarm probability lower than or equal to  $\alpha$ , if*

$$f(u_k) > \gamma_{\min}.$$

□

## 4.8 Asymptotic correctness

In this section, it is presented a set of relevant results regarding the correctness of the SVOs, i.e., the SVOs ability to estimate without error the state of the system. These results allow us to have finite-time consensus even for the case where a node estimates are built using its own local measurements and without receiving estimates from its neighbors. In the next theorem, we show an important feature of the proposed algorithm when applied to fault detection in networks, although its verification may be costly in terms of required computational power.

Before stating the theorems, take a 5-node network as an example,  $n_x = 5$ , where node 1 is running the SVO and has as neighbors nodes 2 and 3. Nodes 4 and 5 are neighbors of nodes 2 and 3. After some time, node 1 will determine exactly nodes 2 and 3 due to direct communication. However, since nodes 4 and 5 are both neighbors of nodes 2 and 3, even though the numeric value for the state of node 4 and 5 can be computed, node 1 cannot associate which numeric value corresponds to which node. The same reasoning allowed to discard edges of the communication graph in Proposition 4.3. Thus, if the true state after some time is  $x(k) = [1 \ 2 \ 3 \ 4 \ 5]^T$ , then  $X(k) = \{[1 \ 2 \ 3 \ 4 \ 5]^T, [1 \ 2 \ 3 \ 5 \ 4]^T\}$ . However, the ordering of the nodes is irrelevant to consensus and the final value can be computed by averaging any of the points in  $X(k)$ .

Following the example, we introduce the nomenclature of permutation matrix as one having one entry equal to 1 in each row and each column, and all the remaining equal to zero. Let us define a set of permutations matrices,  $\mathcal{P}$ , such that all matrices  $P \in \mathcal{P}$  are permutation matrices

with the  $i$ th row (node running the SVO) and  $j$ th row (neighbors of  $i$ ) equal to the respective row in the identity matrix. Also, let  $X^\star := \{Px_{\text{true}} : P \in \mathcal{P}\}$ , where  $x_{\text{true}}$  is the final state of the system. Using  $X^\star$ , we can state the following theorem.

**Theorem 4.3.** *Consider the fault detection described in Section 4.4 where an SVO estimates the state without sharing node measurements and a horizon  $N$ . Take  $X(N)$  constructed using (5.3). Then,*

$$\text{Prob}[X(N) \rightarrow X^\star] \rightarrow 1 \text{ as } N \rightarrow \infty$$

□

*Proof.* Let us rewrite the matrix in (5.3) recursively:

$$\underbrace{\begin{bmatrix} I & -\mathcal{A}_1 & 0 & \cdots & \cdots & 0 \\ -I & \mathcal{A}_1 & \ddots & 0 & \cdots & 0 \\ I & 0 & \ddots & \ddots & 0 & \vdots \\ -I & \vdots & 0 & \ddots & \ddots & 0 \\ I & 0 & \cdots & 0 & \ddots & -\mathcal{A}_{k+1} \\ -I & 0 & \cdots & \cdots & 0 & \mathcal{A}_{k+1} \\ C(k+1) & 0 & \cdots & \cdots & \cdots & 0 \\ -C(k+1) & \ddots & 0 & \cdots & \cdots & 0 \\ 0 & \ddots & \ddots & 0 & \vdots & \vdots \\ 0 & 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & 0 & \ddots & C(1) & 0 \\ 0 & \vdots & \ddots & 0 & -C(1) & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & M_0 \end{bmatrix}}_{\mathbf{M}_{\Delta^\star}} \begin{bmatrix} \mathbf{x}_{k+1} \\ \mathbf{x}_k \\ \vdots \\ \mathbf{x}_0 \end{bmatrix} \leq \begin{bmatrix} 0 \\ \vdots \\ 0 \\ y(k+1) \\ -y(k+1) \\ \vdots \\ y(1) \\ -y(1) \\ -m_0 \end{bmatrix} \quad (4.18)$$

where  $\mathcal{A}_n$  represents the matrix  $A_0 + A_{\Delta^\star}$  with a  $\Delta^\star$  that accumulates the uncertainties for  $n$  periods of time, i.e., the parameter  $\Delta^\star$  is the uncertainty instantiation for the respective horizon (see [RS13]).

Consider node  $i$  is running an SVO and define a sequence of time instants in an iterative fashion, as follows:

- $\exists k \in [0; k_1^\star]$  where there exists a communication between  $i$  and all of its first degree neighbors where only the state is transmitted and not the estimates, i.e.,  $\forall j : (i, j) \in E$  we have  $A(k) = Q_{ij} \vee A(k) = Q_{ji}$ ;
- $\exists k \in ]k_1^\star; k_2^\star]$  for all second-degree neighbors where there exists a communication between that node and the neighbor of  $i$  followed by a communication from the neighbor to node  $i$  itself, i.e.,  $\forall j : (i, j) \in E, \forall \ell : (j, \ell) \in E$  we have  $A(k) = Q_{j\ell} \vee A(k) = Q_{\ell j}$  and  $A(k+1) = Q_{ij} \vee A(k+1) = Q_{ji}$  and ;

- repeat the same as before, for the third-degree neighbors, where transmissions occur in the interval  $[k_2^*; k_3^*]$  with communication between the nodes happening at each multiple of three communication instants. The number of communications must be equal to the number of possible paths with length 2.
- we continue with the same reasoning until all the nodes are included in the sequence.

Since when a node is involved in a communication there is no uncertainty, the sequence was constructed such that with the first condition all the neighbor states can be determined. With the second condition all the second-degree neighbor states can be determined. The same applies for any degree neighbor. This implies that for a specific instantiation of  $\Delta^*$ , the system in (4.18) either:

- has only one solution;
- is infeasible.

Thus, the estimate set  $X(k)$  is a union of at most  $\text{card}(\Delta)$  points.  $\forall \epsilon > 0, \exists N$  such that the sequence exists with probability  $1 - \epsilon$  and the conclusion follows.  $\square$

The previous result shows that SVOs have an intrinsic correctness property that can be used to compute the average consensus. Theorem 4.3 assumes that estimates are not shared between neighbors at the expenses of considering a large horizon  $N$ . Nevertheless, in practice its applicability is questionable, as  $N$  can be arbitrarily large and represent a prohibitive computational burden. Since the SVO complexity grows exponentially with the horizon, one cannot use Theorem 4.3 to determine the states of each node in the network, in the general case. However, the result is interesting in the scenario where the node running the SVO is controlling the network and is allowed to impose a given communication pattern. In such cases, it can calculate a pattern ensuring the conditions of the theorem are fulfilled, guaranteeing finite-time consensus and detection of (detectable) faults in the sense of Definition 4.1. Progress is made in the next theorem to drop the horizon condition by taking advantage of state sharing between nodes.

**Theorem 4.4.** *Consider the algorithm described in Section 4.6 and illustrated in Figure 4.3 and  $X(\tilde{N})$  constructed using (4.7). Then,*

$$\text{Prob}\left[X(\tilde{N}) \rightarrow \{x_{\text{true}}\}\right] \rightarrow 1 \text{ as } \tilde{N} \rightarrow \infty$$

$\square$

*Proof.* Construct the sequence of time instants  $\{c_k : 0 \leq c_k \leq \tilde{N}\}$  that fulfills the following conditions

- every transmission shares one of the nodes involved in the previous transmission, i.e.,

$$\forall k \in \{c_k\} : A(k) = Q_{ij}, A(k+1) = Q_{i\ell} \vee A(k+1) = Q_{\ell i}$$

for any node  $\ell$ ;

- there exists a time instant such that before and after that time, all the nodes in the network were involved in the communication, i.e.,

$$\exists k_c \forall i \exists k_i \leq k_c : (A(k_i) = Q_{i\ell} \vee A(k_i) = Q_{\ell i})$$

$\wedge$

$$\exists k'_i \geq k_c : (A(k'_i) = Q_{i\ell} \vee A(k'_i) = Q_{\ell i})$$

for any node  $\ell$ .

$\forall \epsilon > 0, \exists N^*$  such that this sequence exists with probability  $1 - \epsilon$ .

Define a function

$$V(k) = \text{card}(\tilde{z}_i(k))$$

where the function  $\text{card}(x)$  counts the number of non-zero entries of vector  $x$ , and  $i$  is a node involved in communication at time  $k$ . Function  $V(k)$  counts, therefore, the number of uncertain states of the last node  $i$  involved in a communication at time  $k$ , and

$$\tilde{z}_i(k) = [z_i(k)]_{2i-1} - [z_i(k)]_{2i}.$$

Recall that, from equation (4.13), both nodes  $i$  and  $j$  involved in the communication have the same estimates of the states for all the nodes in the network.

Moreover, notice that

$$V(k+1) - V(k) \leq 0$$

for all time instants  $k \leq k_c$ , since every transmission is assumed to include one node involved in the previous communication and it is a strict inequality whenever it is the first time the node appears in a communication. In addition, the equilibrium points satisfy  $\text{card}(\tilde{z}_i(k)) = 0, \forall i$  by construction, since they are the only points that, when computing the new set-valued state estimates, will return a set with only one point. Thus, for some time  $k_c \geq 0$ ,  $V(k_c) = 0$  using the two conditions of the sequence, which means that the two nodes communicating at time  $k_c$  have access to the full state of the network, regardless of the horizon of the SVOs. By the discrete version of the La Salle Principle, the conclusion follows.

Since, for every node  $\ell$ ,  $\exists k'_i \geq k_c : A(k'_i) = Q_{i\ell}$ , the full state is passed to all the remaining nodes. We conclude that all nodes have  $X(k)$  equal to a singleton.  $\square$

**Remark 4.2.** Notice that, in practice, by implementing a token-passing scheme, the algorithm can be forced to converge in finite-time regardless of the chosen horizon, if no fault is detected.



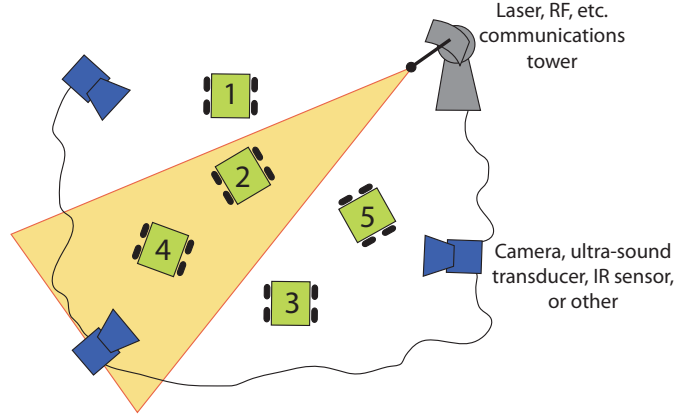


Figure 4.4: Illustrative example of the setup for the problem.

The main point of the construction was that any two consecutive time instants share one of the nodes that communicated. However, caution is necessary to avoid reducing the algorithm to a deterministic setting. One possible solution is to consider that the token is passed randomly when communicating (i.e., with a probability  $p$ , the node sends the token when it communicates, and with probability  $1 - p$  the node retains the token). In addition, instead of nodes having equal probability  $\frac{1}{n_x}$  of initiating a communication, the probability distribution is concentrated in the node that possesses the token. This means that there is a non-zero probability of a node starting a communication even though it does not possess the token.

The advantage of having a non-zero probability for any node to initiate a communication is to prevent an attacker from stopping the whole network by controlling the node that possesses the token. Mechanisms for fault robustness in a token-based gossip algorithm are outside the scope of this chapter and also further work is needed to evaluate its effects on the convergence rate.

## 4.9 Application of Set Estimators to Set Consensus

The consensus problem described in Chapter 2 assumed the state of the nodes can be measured without any noise and that the dynamics are not perturbed by disturbances. If that is not the case, instead of a single point for the state, the nodes can compute a set containing the true state. From the discussion in this chapter, it follows that set estimators are an alternative to deal with such problems. In [SHGE14], the authors address the former issue, but specify a particular shape for the sets and assume all-to-all communication. This section applies the aforementioned techniques to the set-consensus problem.

In more detail, this section addresses the problem of having  $n$  robots or agents that are trying to reach consensus over their positions. Due to desired costs savings or environment constraints, the robots are only equipped with receivers and have no sensing and self-localization capabilities. A tower takes measurements of the position and velocity of each robot by using, for example, a

vision-based system, and uses directional antennae to forward that information to the nodes. To avoid the computational cost of maintaining sets for all the positions and velocities of the nodes in the network and have a solution suitable for large scale networks, we look to solve the problem in a distributed fashion. An illustration of the problem is depicted in Figure 4.4. It is remarked that, although we will analyze this particular setup, the problem formulation can be extended to a myriad of other scenarios. For example, a sensor network where each node decides by itself when to take measurements of the state of the network and then sends to nearby neighbors. In such case, both the measurements are noisy and are performed at different time instants and need to be updated.

In the aforementioned setup, each node  $i$  will receive a set  $X_j$  for each of their neighbors  $j$  corresponding to the position and velocity with the corresponding measurement errors and possible disturbances. In our context, neighbors refer to nodes that are sufficiently close so as to belong to the same strip of field to which the central tower communicates. The tower defines  $m$  partitions of the terrain in such a way to cover the whole space where nodes can be. However, sensing and communication is not performed all at the same time instants, and thus node  $i$  might receive, for example,  $X_1(k-3)$ ,  $X_2(k-1)$  and  $X_3(k)$  as a result of receiving the data destined to nearby nodes.

We consider the dynamics of each robot to be described by a Linear Parameter-Varying (LPV) model  $S_i$  of the form:

$$S_i : \begin{cases} x_i(k+1) = A_i(k)x_i(k) + B_i(k)u_i(k) + E_i(k)d_i(k) \\ y_i(k) = C_i(k)x_i(k) + n(k) \end{cases} \quad (4.19)$$

The signal  $u_i$  is the actuation signal that will be used to find consensus using the sets of variables of other robots in the vicinity. Matrices  $A_i(k)$  in (4.19) are the sum of a single central matrix  $A_0$  with parameter-dependent terms as in (4.4). In the context of this problem, the parameter  $\Delta$  can model uncertainties in the mass of the robots. For example, if their task is picking objects across a field, the mass of each robot might be uncertain depending on which objects and corresponding masses were picked.

Since the measurements received by a node refer to different time instants, it is necessary to propagate the states to the current time. Using the framework of SVOs, the estimate sets received by the central unit can be denoted by  $X(\cdot)$ , whereas the propagated sets to a single time instant by  $\tilde{X}(\cdot)$ . The updated estimates are subject to an approximation since the dynamics matrices have uncertainties. Thus, the use of the symbol  $\sim$  distinguishes between an approximation using the procedure for SVOs for uncertain models or the exact estimate provided by the centralized tower.

In our context, the SVO is going to be initialized with the measurement received from the central tower and the corresponding set for a particular agent is computed for the current time instant. By doing so, each node is going to construct a set-valued estimate of the position and velocity of each node at the present time.

In the following subsections, two different cases are used in this context that illustrate how the SVOs can be used in a centralized and distributed fashion.

#### 4.9.1 Broadcast solution using position

We start by looking at the problem of when the measuring sensors track each of the agents in the desired angle of communication and then forward a message containing this information. In this case, each node in that strip will receive, at the same time instant, the position and velocity measurements for all its neighbors, which, however, may have been taken at different time instants. In such a setup, all the neighboring nodes have the same information and will perform the same tasks.

A possible solution is to have each node taking the Minkowski sum (i.e.,  $X + Y$  denotes the set of vectors  $z \in \mathbb{R}^n$  such that  $z = x + y$  for  $x \in X$  and  $y \in Y$ ) of the received position sets and calculates which actuation it should use to drive itself to that position.

The new position set is given by

$$\tilde{X}_i(k+1) = \alpha \tilde{X}_i(k) + (1-\alpha) \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} \tilde{X}_j(k) \quad (4.20)$$

where the parameter  $\alpha$  is used to model a possible drawback from having node  $i$  changing too much its position. Each node can have different values for  $\alpha$  to reflect their diversity. We stress again that the sets  $\tilde{X}_j(k)$  are built using the SVO update scheme from the sets  $X_j(k - k_j)$  that were received and which correspond to the position and velocity estimates of the agents in the vicinity.

The actual control law can be found by computing the translation that better changes  $\tilde{X}_i(k)$  to fit  $\tilde{X}_i(k+1)$ . This resorts to solving an optimization problem to find such control input, i.e.,

$$\begin{aligned} v = \arg \min \quad & \max_{x,y} (\|(v+x) - y\|) \\ \text{subject to} \quad & x \in \tilde{X}_i(k) \\ & y \in \tilde{X}_i(k+1), \end{aligned} \quad (4.21)$$

where  $\tilde{X}_i(k+1)$  is defined as in (4.20).

Our optimization variable is  $v$ , the translation vector, which is equivalent to the velocity vector that is needed to drive the system from where it is at the present time instant, to the weighted average of the set-valued positions of the remaining nodes in the vicinity of the node.

Alternatively, one could also solve the problem in a slightly different setting by focusing on reducing the distance of the node position to that of its neighbors. The problem would be rewritten as

$$\begin{aligned} v = \arg \min \quad & \max_{x,y} \left( \sum_j \|(v+x) - y_j\| \right) \\ \text{subject to} \quad & x \in \tilde{X}_i(k) \\ & y_j \in \tilde{X}_j(k). \end{aligned} \quad (4.22)$$

The focus of this work is not on addressing this issue, but it is stressed that one possible approach to this problem is to find the circumference that best fits each of the polygon in two dimensions, and then use their centers to compute the translation vector. Depending on the agent dynamics, the control input will be different but, essentially, aims at driving the position according to the vector  $v$ . The whole algorithm can be summarized in Algorithm 2.

---

**Algorithm 2** Set-consensus without position estimation

---

**Require:** Sets  $X_j(k - k_j)$ .

**Ensure:** Computation of the velocity to be applied.

```
1: for each  $j$  do
2:   /* Update sets  $X_j(k - k_j)$  to get  $\tilde{X}_j(k)$  */
3:    $\tilde{X}_j(k) = \text{update\_SVO}(X_j(k - k_j))$ 
4: end for
5: /* Find translation vector  $v$  */
6:  $v = v^*$  where  $v^*$  is found using (4.21) or (4.22)
7: /* Compute  $u$  */
8:  $u(k) = v$ 
```

---

### 4.9.2 Unicast solution using estimation

In the previous subsection, the setup where nodes receive all the information in their vicinity was discussed. Since all nodes in a strip receive the estimates for all remaining nodes, it makes possible to determine their velocities by computing the destination as a Minkowski sum of all the estimates. In this subsection, we consider a different setting where the tower unicasts messages with the information of a single agent. However, due to the shared medium, their neighbors are able to sense and receive those communications. In such a setup, the destination of the message is unaware of their neighbors, but their neighbors discover their presence since they also receive the message.

The proposed solution is to augment the state of the SVOs with the states of the neighbors. The set  $\tilde{X}_i(k)$  are updated using the same SVO tools, but considering the information received as observations of the whole system with states as the concatenation of positions  $x_i$  and  $x_j, j \in \mathcal{N}_i$ , i.e., all the neighbors of node  $i$ .

In order to take into account the possible actions that neighboring nodes take as the result of receiving information for their own neighbors, we use a disturbance term as in (4.19). The new definition for the set  $\tilde{X}_i$  means that, before calculating the control input, we need to project  $\tilde{X}_i$  on the  $i$ th coordinate to obtain the set-valued estimate for node  $i$  position and discard the positions of the neighboring nodes.

The new algorithm is as described in Algorithm 3.

---

**Algorithm 3** Set-consensus with position estimation
 

---

**Require:** Sets  $X_j(k - k_j)$ .

**Ensure:** Computation of the velocity to be applied.

```

1: for each  $j$  do
2:     /* Construct set  $\tilde{X}_i(k)$  */
3:     Add an observation to (4.8) corresponding to  $X_j(k - k_j)$ 
4: end for
5:  $\tilde{X}_i(k) = \text{update\_SVO}()$  using (4.8)
6:  $\tilde{X}_i(k) = \text{projects}(\tilde{X}_i(k), i)$ 
7: /* Find translation vector  $v$  */
8:  $v = v^*$  where  $v^*$  is found using (4.21) or (4.22)
9: /* Compute  $u$  */
10:  $u(k) = f(v)$ 
    
```

---

### 4.9.3 Convergence to Set-consensus

In this subsection, we present a convergence result for the proposed algorithm which ensures that all the nodes converge to a cluster, where the distance among themselves depends on the size (or uncertainty) of the sets, as described in the sequel. We define an overbounding ball of radius  $\epsilon$  and center  $c$  to be denoted as  $B_\epsilon(c)$ .

**Theorem 4.5.** *Take  $n$  nodes running Algorithm 3 and define  $\epsilon_{\max}$  such that  $\exists c_i, i \leq n : \forall k, \tilde{X}_i(k) \subset B_{\epsilon_{\max}}(c_i)$ . Then, all of the nodes converge to at most  $m$  clusters, where each of these clusters is defined as a neighborhood  $2\epsilon_{\max}$ , i.e., for a given center  $c_g, g \leq m$  we have that  $\exists g$ :*

$$\forall i : \lim_{k \rightarrow \infty} x_i(k) \in B_{\epsilon_{\max}}(c_g)$$

*Proof.* Let us start by using the assumption that there is an overbounding ball at all times for the sets  $\tilde{X}_i(k)$ , which means that we can study the convergence of  $B_{\epsilon_{\max}}(c_i(k))$  instead of the sets  $\tilde{X}_i(k)$ , where we made explicit that the center varies with time.

Notice that the control input  $u(k)$  is only going to shift  $c_i(k)$ , which means that we can focus on determining if the centers of the bounding balls are converging.

Let us define a Lyapunov function for the evolution of the centers of the overbounding balls

$$V(k) = \max_{i,j} \|c_i(k) - c_j(k)\|$$

which obviously is bounded below since the distance cannot be negative. Take node  $i$  to be the one with the largest  $x$  coordinate and  $j$  to be the one with the smallest (the same reasoning applies to the  $y$  coordinate). From solving the optimization problem (4.22),  $[c_i(k+1)]_x \leq [c_i(k)]_x$  and  $[c_j(k+1)]_x \geq [c_j(k)]_x$  since both nodes minimize their distance to the remaining nodes. Thus,

$$V(k+1) \leq V(k),$$

and the inequality is only strict if the nodes  $i$  and  $j$  belong to the same neighborhood. Therefore, if we divide the analysis for each of the strips of terrain covered by the antennae, we get the

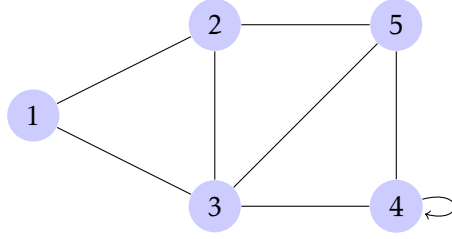


Figure 4.5: Communication graph used for simulation.

strict inequality, meaning that  $V(k)$  is monotonically decreasing. In addition,  $V(k) > 0$  except when all the centers are equal, in which case,  $V(k+1) = V(k)$ . By the discrete-time version of the La Salle Invariance Principle, the centers are all converging to a common value as  $k \rightarrow \infty$ , at which point,  $\max_{i,j} \|x_i(k) - x_j(k)\| = 2\epsilon_{\max}$ , thus concluding the proof.  $\square$

The previous result states that the convergence for a static partition of the field is only going to yield the formation of  $m$  clusters, where  $m$  is exactly the number of partitions. The conclusion is derived from the fact that the centers of the polytopes are all converging to a weighted average of their centers and, therefore, away from the limits of the partitions. We can see this result as the convergence of a consensus algorithm for a partitioned connectivity graph.

In Section 4.10, we will use a simple setup to show through simulations that varying the partitioning method to a simple round-robin along the two dimension of the ground yields convergence to a single cluster, i.e., Theorem 4.5 is satisfied with  $m = 1$ .

## 4.10 Simulation Results

In this section, we show simulation results for some meaningful scenarios which are used to illustrate specific features of the proposed fault detection schemes: deterministic, stochastic and consensus algorithm with fault detection. Two different types of faults are tested against the standard deterministic SVO when running in a single node. Comparison is also made to the case where each node runs a local SVO as to determine the first time of detection. A third type is detected by the SSVO, to motivate the use of the stochastic information, when a worst-case detection is not suitable. Lastly, the properties of the consensus algorithm are demonstrated, in particular its finite-time convergence.

The network used in the simulations has a small number of connections between the node computing the estimates and the remaining nodes as to make the detection more challenging. The intuition is that the node computing the estimates will not directly observe all the nodes making the detection harder. Without loss of generality, we illustrate the results from the perspective of node one with a faulty neighbor, i.e., the output  $y(k)$  corresponds to the observations of one of the neighbors of the faulty node.

We consider a 5-node network with nodes labeled  $i, i \in \{1, 2, 3, 4, 5\}$  and initial state  $x_i(0) = i - 1$  and a nominal bound for the state magnitude of  $|x_i| \leq 5$ . In order to reduce complexity and

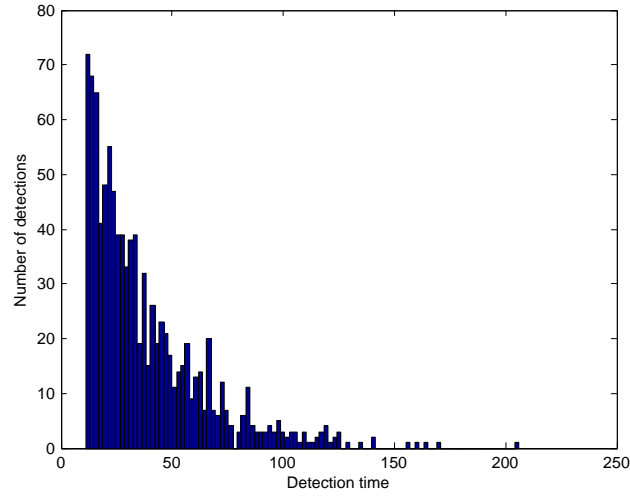


Figure 4.6: Detection times for the stochastic fault.

to study the properties of the algorithms in a disadvantageous setting, we considered  $N = 1$ , meaning that we only use the information from the previous iteration for the estimates. This is a worst-case scenario, as the algorithm only takes into account the dynamics of the system with one time step from the last estimate and discards prior observations and their propagation using multiple steps with the system dynamics. A missed detection is considered if the algorithm is not able to detect the fault within 300 observations. Each result presented corresponds to 1000 Monte-Carlo runs. For convenience, node 1 is the node that performs the detection and node 2 is the failing node, and no faults occur in the first 10 transmissions. Note that if a node sends a different value than its initial state from the start of the simulation, it can trivially do so without being detected since the network has no information about the initial state of that node. The following probability matrix is used:

$$W = \begin{bmatrix} 0 & 0.5 & 0.5 & 0 & 0 \\ 0.5 & 0 & 0.25 & 0 & 0.25 \\ 0.5 & 0.25 & 0 & 0.125 & 0.125 \\ 0 & 0 & 0.125 & 0.25 & 0.625 \\ 0 & 0.25 & 0.125 & 0.625 & 0 \end{bmatrix}$$

The first scenario corresponds to an erratic node failure in which the node will respond with a random value. Specifically, after 10 iterations the node always replies as if its state was drawn uniformly from the interval of admissible states  $[-5, 5]$ .

Figure 4.6 depicts the histogram of the detection times for the aforementioned fault. In this simulation, the detection rate was 100%, which is not surprising from the erratic behavior of the node. Analyzing the distribution, one key observation that is recurrent in other simulations is that, as time passes, the detection is more likely to occur. At the moment of detection, we have  $\gamma_{min} = 56.25$  and the correspondent magnitude of the injected signal  $\|u\|^2 = 4.405$ . We

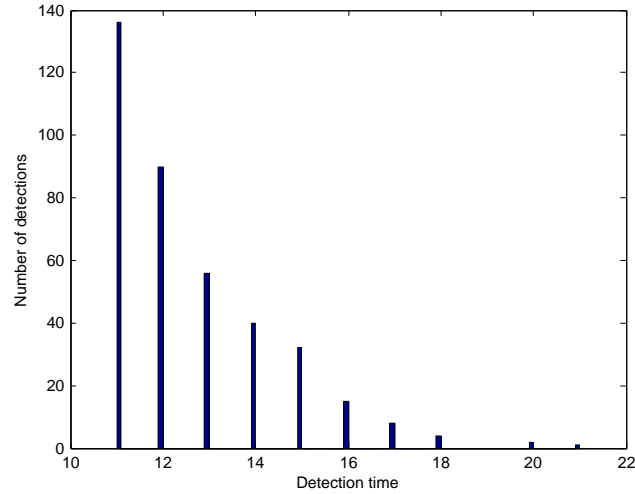


Figure 4.7: Detection times for the deterministic fault.

concluded that the value of  $\gamma_{min}$  as a worst-case scenario is conservative in the sense that signals with a smaller energy are also detected.

We also considered a less erratic scenario where a node becomes unresponsive due to CPU load or software crash, does not perform the consensus update and, therefore, replies always with the same value.

Figure 4.7 depicts the detection time for the deterministic fault where the node replies with the same value. In this case, the detection rate is 38.4%. In some sense, the lower detection rate is motivated by the fact that this fault does not change the state as much as the previous one. Since node 2 has other neighbors not in common with node 1, the fault is undetectable in more transmission sequences than in the previous simulation. Nonetheless, we still observe the behavior that the fault is more likely to be detected as time progresses. Once again, we calculate  $\gamma_{min} = 76.56$  and  $\|u\|^2 = 2.997$  and observe that the injected signal is still detected even though its energy is less than the theoretical bound.

To illustrate the benefits of the SSVO when detecting faults, we consider a scenario where a node takes advantage of the network and initiates communication with a neighbor regardless of the probability matrix  $W$ , but does not change any of the nodes state. Notice that using an SVO, such faults would not be detected as any communication pattern that is possible is considered, regardless of its probability. Between transmission time  $10 < k < 20$ , it is assumed that the communication takes place between node 3 and 4. Moreover, define  $\alpha = 0.1$ .

Figure 4.8 depicts the detection times for the SSVO case with a detection rate of 92.8%. Even though the behavior is still the same, we can no longer guarantee that the detection is caused by the fault and not by a communication pattern which we consider to be a fault, but that has non-zero probability of occurring in a healthy scenario.



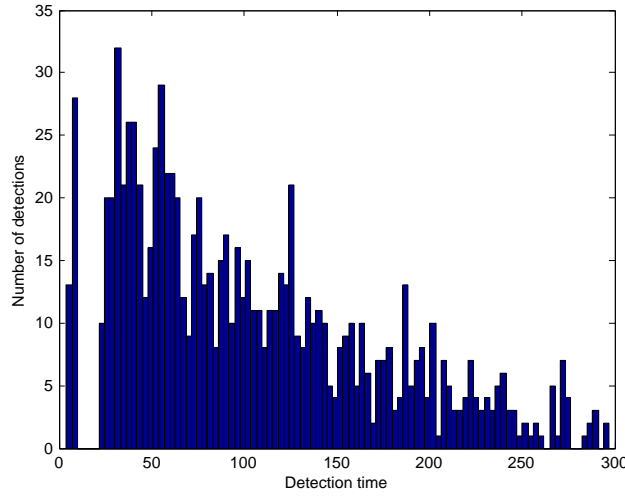


Figure 4.8: Detection times for the SSVO.

In the previous simulation results, we depicted the detection time for a single-node point of view in the network. However, when running the detection scheme presented in this chapter, each node will run an SVO of their own to estimate the possible set of states and it is therefore important to assess the first time any node detects the fault. The simulation setup is the same as before and we assume that a node is trying to drive the consensus value by repeating the same value. Without a fault detection scheme, all the nodes would asymptotically reach a final consensus equal to the repeated constant. To make the results comparable, the data presented was generated using a thousand different seeds for the random number generator used to select the communication pairs, according to the probability matrix  $W$ .

In Figure 4.9, the average difference between the time that any node detects a fault and that node 1 detects the fault is presented. For a horizon equal to 1, we have a huge difference motivated by the fact that when considering just the detection from node 1, and using this faulty scenario, there is a remarkable number of undetected faults leading to considering the detection time as 300 time steps, which is the maximum length of the simulation. For the remaining values of the horizon, we have an increase in the detection time, which illustrates the importance of considering the different observations available to the nodes.

Another interesting issue is to determine the impact of changing the horizon in the detection time. By construction, incrementing the horizon leads to a smaller or equal time of detection. The rate at which the detection time varies is of particular interest when assessing the trade-off

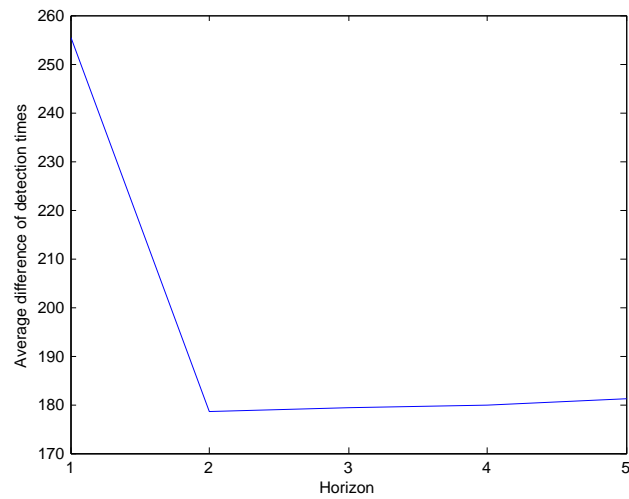


Figure 4.9: Average difference between detecting with a SVO in one node or in all the nodes.

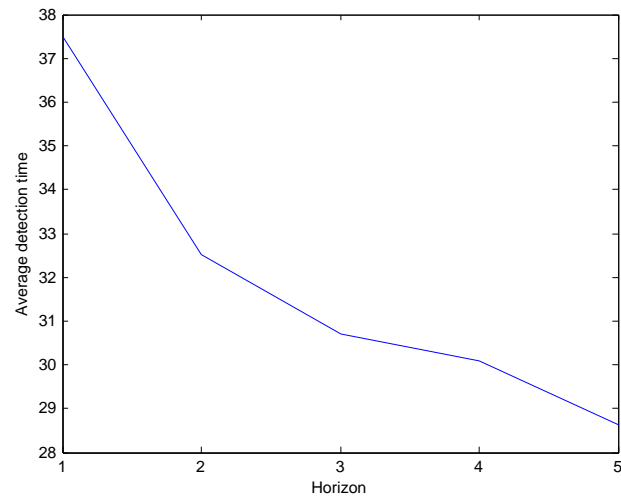


Figure 4.10: Detection time for different horizon values for a fault constant equal to 3.

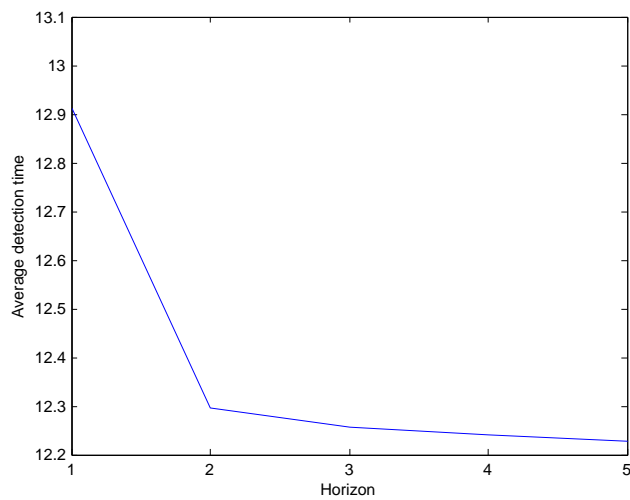


Figure 4.11: Detection time for different horizon values for a fault constant equal to 4.9.

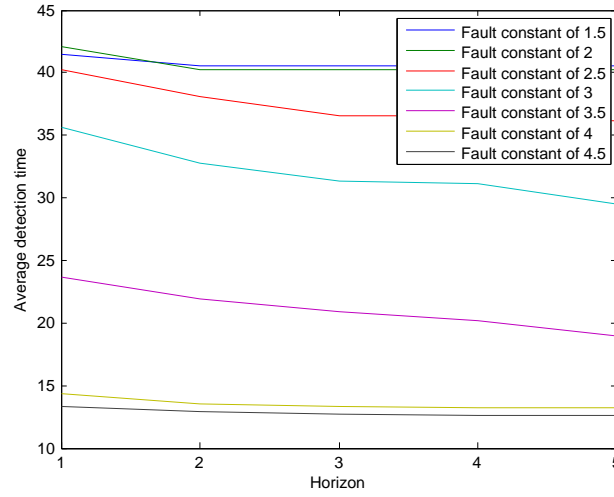


Figure 4.12: Detection time for different fault constants.

between fast detection and computational complexity.

In order to show the decreasing trend in the detection time as the horizon increases, we selected two fault constant values, namely 3 and 4.9. The intuition behind this choice is that a fault characterized by using a constant 4.9 is “easier” to detect, since the magnitude of the difference between the constant and the true state is larger than when considering a fault constant of 3. Figure 4.10 and Figure 4.11 show the mean detection time for different horizon values of having a fault constant equal to 3 and 4.9, respectively. When considering the case of constant 3, there is a faster decrease in the detection time which goes from over 37 time steps when the horizon is equal to 1, to under 29 when the horizon is equal to 5. For the case of constant 4.9, the difference is between using a horizon equal to 1 and higher horizons.

Emphasizing on the observed behavior, we present in Figure 4.12 the mean detection time for different constant values. From Proposition 4.4, this phenomenon can be seen as the magnitude of the fault approaching  $\gamma_{min}$ , which is the worst case for the magnitude of the injected signal before being detectable in the worst case scenario. Depending on the specific application, the horizon can be selected so as to meet the specific requirements. In the example of consensus, the horizon can be selected in order to decrease the expected deviation in the final consensus value, since by increasing the horizon, the maximum magnitude of the input signal decreases. In applications where the computation cost is not a problem, but there is a demanding criteria for the detection time, the horizon should be set as close to  $N^*$  as possible. However, for real-time applications, where the running time of the detection is crucial, a small horizon should be selected and the detection scheme becomes a best-effort approach.

We now present simulations that illustrate the finite-time consensus property derived in the previous section. Focus is given on how a measure of the set dimension evolves with the algorithm as opposed to a setting where nodes just run SVOs without sharing their estimates. The simulations also indicate how likely it is to find a sequence of transmissions that produce

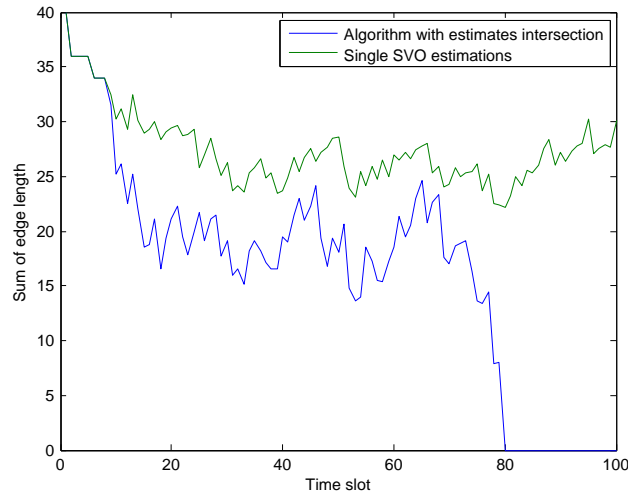


Figure 4.13: Typical behavior of the size of the SVO.

finite-time consensus when using randomized gossip algorithms.

Our experiment setting for the following tests does not include any fault and, at each time instant, we compute a measure of the size of the SVO. Computing the volume would be meaningless since at least the dimension corresponding to the node value has size zero, as the node has access to its own value at all time. Since the representation of the set of estimates is converted into a hyper-parallelepiped before being sent to a neighbor upon communication, we sum the length of uncertainty for each state and regard that measure as the size of the set. Each node has its own set-valued estimate, which we represented as a vector after bounding with a hyper-parallelepiped, as described in the previous section. For that reason, to measure the size of the SVOs across the network, we take the mean values (computed element-wise) of those vectors. By definition, if such measure reaches zero, then all nodes have reached consensus.

Figure 4.13 depicts a typical run where finite-time consensus is achieved. All the simulations share the same behavior and what distinguishes them is the time when consensus is achieved for the algorithm. In comparison, the same measure is calculated for the case where each node runs its own independent SVO computed using only its own measurements. As expected, the estimates using the algorithm are less conservative as they incorporate the measurements performed by the node itself and the estimation set transmitted by its neighbors. In this particular run, consensus was achieved by all the nodes at iteration 80.

Using a 1000 Monte-Carlo run, in Figure 4.14 is shown the histogram for the stopping time of the algorithm when using a horizon of 1. The experiments where consensus was not achieved in less than 300 communications are not represented in the histogram and corresponded to 21.9% of the cases. We then repeated the simulations for the same sequence of communications using a horizon of 5. The percentage of experiments that did not end in a finite-time consensus within the 300 time instants were 13.4%. The decrease is justified by the smaller sets that each

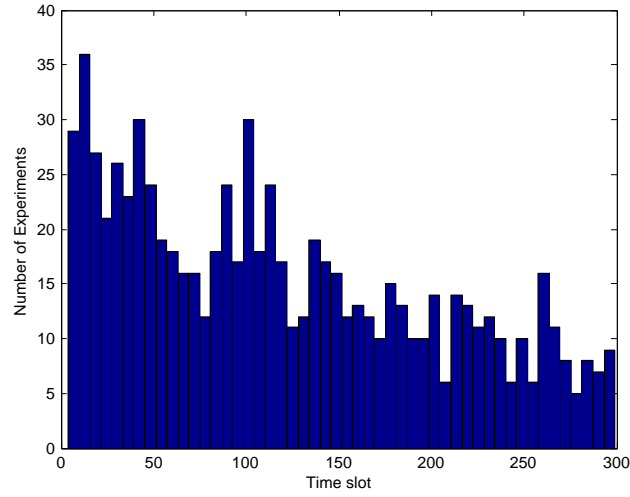


Figure 4.14: Histogram for the stopping time with the proposed algorithm.

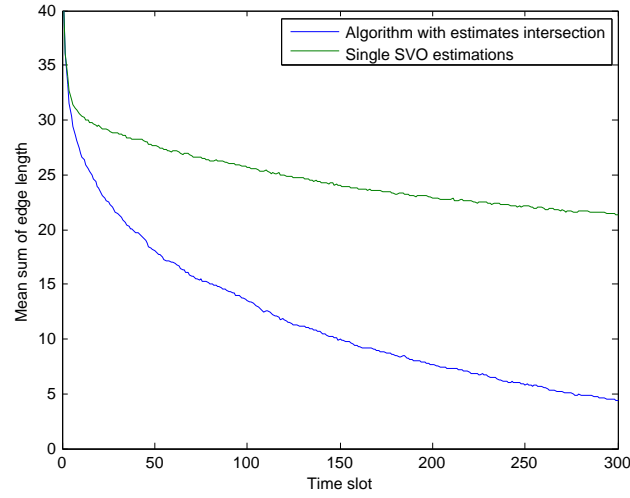


Figure 4.15: Evolution of the mean sum of edges of all node set-valued state estimations.

node generates. In essence, to get 100% of the experiments to end in finite-time, we either have to increase the time of the simulation, increase the horizon, or both.

An important issue is the influence of the intersection step on the size of the set-valued state estimates. Figure 4.15 depicts the mean of the sum of edges length for the 1000 Monte-Carlo runs for both the case of an SVO with and without estimate sharing using the intersection algorithm. Since the gossip random consensus algorithm is stable [BGPS06], the size of the generated set converges to a point (the consensus value) and the sum of edge lengths goes to zero asymptotically when in a non-faulty scenario and subject to a horizon smaller than  $N^*$ . The measure of the sum of edges captures the size of the set-valued estimates, and correspondingly, how conservative they are. Figure 4.15 shows that, in average, estimates are less conservative by exchanging set-valued estimates. Also, the set-valued state estimates, provided by the proposed algorithm, converge much faster to zero, since the conditions of the Theorem 4.4 are less

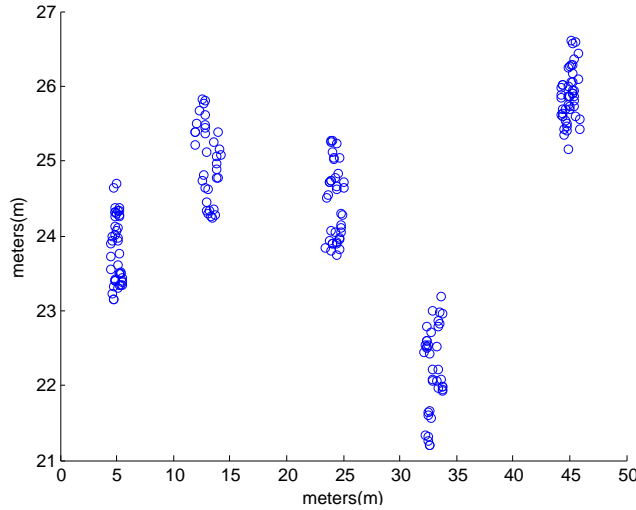


Figure 4.16: Final distribution of the nodes after 100 time instants using one antenna.

restrictive.

The application of SVOs to other setups was also illustrated with the setup for set-consensus. We performed some simulations to show the convergence of the true positions of the agents when only a set-valued measurement is available that is guaranteed to contain the true state. In particular, we look at a simple round-robin policy to make the nodes converge to a single cluster instead of  $m$  clusters depending on the number of partitions for the ground.

The simulations considered 200 nodes randomly distributed across a 50m×50m square field with an antennae mounted on both sides. Nodes will receive the information transmitted and move according to the proposed algorithm but will only have access to the set-valued estimates of their positions, and not the true (noise-free) positions.

We consider two different scenarios: one where only one of the antennae is functioning and dividing the field into 10 partitions along the  $x$  coordinate, going in a round robin fashion over them and using an offset value to cover different ground strips at each time; and a second example, where two antennae with the arrangement to be described next alternate every 5 time instants.

Figure 4.16 depicts the final distribution of the 200 nodes for the first case. In this particular run, the number of clusters is  $m = 5$  and it is observed a common behavior where nodes align themselves with the strips of the ground. The reduction of the number of clusters from 10 to 5 is justified by the offset of the transmissions as it increases the connectivity of the network, in the sense that nodes will belong to different clusters in different time instants.

In Figure 4.17, it is shown the evolution of the maximum distance between two nodes in the network over time. This measure illustrates how the performance of the consensus algorithm degrades due to the poor choice of the field stripping. Nevertheless, it is possible to detect when the cluster convergence happened by looking at when the maximum distance between any pair

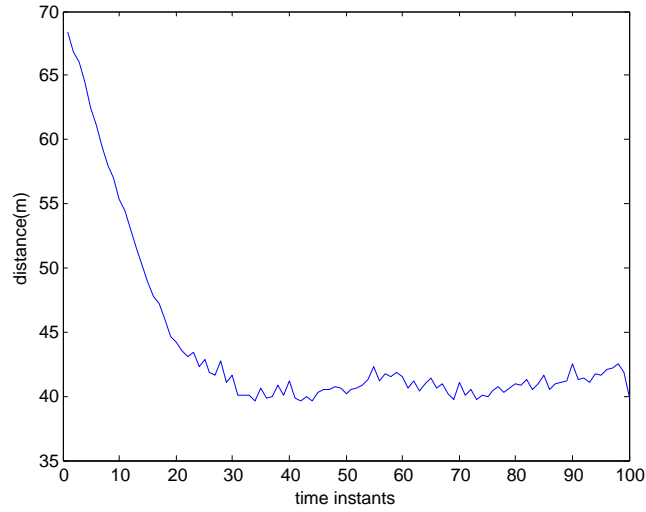


Figure 4.17: Evolution of the maximum distance between two nodes over the 100 time instants of the simulation using one antenna.

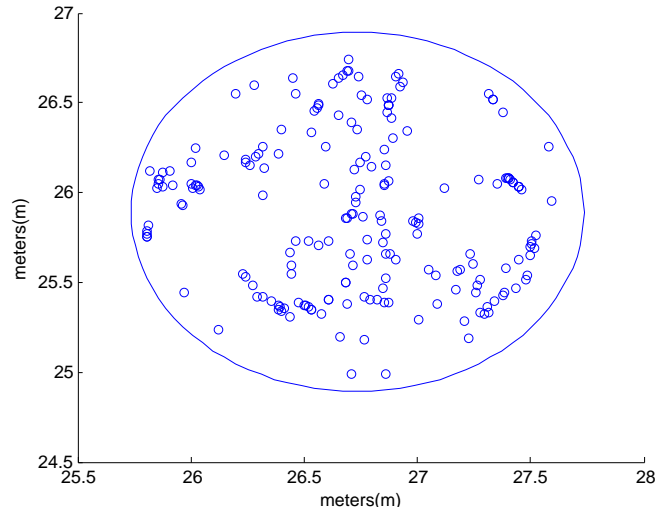


Figure 4.18: Final distribution of the nodes after 100 time instants using both antennae.

of nodes converged.

Based on these results, we introduced the second scenario where the antennae work in alternation. In this setup, an antenna along the horizontal axis and another in the vertical axis transmit in a round robin fashion between them and transmitting in a round robin between their own partitions. The idea is to increase the connectivity and to explore the fact that more nodes will belong to more than one partition over time. The study of different partitioning methods is left as a path for future research.

In Figure 4.18, it is shown the final distribution of the nodes after 100 time instants. We assumed a maximum radius for all measurements of 1 and the ball  $B_{\epsilon_{\max}}(c)$  is shown where  $c$  was computed as the average of the centers for the overbounding balls for each measurement

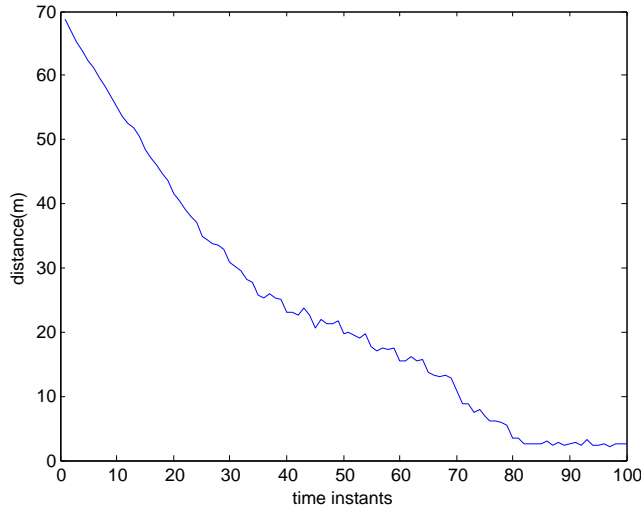


Figure 4.19: Evolution of the maximum distance between two nodes over the 100 time instants of the simulation using both antennae.

set. Following Theorem 4.5, all the nodes converged to a single cluster with radius  $\epsilon_{\max}$ .

In comparison with the previous scenario, we computed the maximum distance between two nodes to have a sense of the convergence rate of the algorithm. We remark that studying this convergence rate is an interesting topic even though its improvement depends on the partition schedule.

When implementing a stopping time for the nodes to declare convergence, a possibility is to consider whether the measurements that they are receiving are similar to that of the remaining neighbors, or if the current set-valued position estimate is close to the final destination of the node.

## 4.11 Conclusions

In this chapter, the problem of fault detection in randomized gossip algorithms is addressed using Set-valued Observers). The introduction of the stochastic information to build the set is one of the main contributions of this chapter that allows to detect faults based on the probability of that event. Two functions to measure the maximum attacker input signal for an undetectable fault are presented. The quadratic function is suitable for systems where the energy plays an important role whereas the linear function is characteristic of problems such as the consensus, where inputting a positive signal cancels the effect of a fault injecting a negative signal. We also showed the necessary number of past observations for the case of local information, when keeping the best value for the horizon is computationally intractable.

Building on the results of having an SVO for fault detection, without sharing state estimates, SVOs in the absence of faults are capable of determining average consensus in finite-time using only measurements available to the node, but may require a large computational burden. The



result is suitable to situations where one node is able to control/command the sequence of communications in the network.

In order to drop the requirement of a large horizon, an algorithm is presented where each node computes its own set-valued state estimates and performs an intersection with state estimates received by the neighbors. Besides reducing the computational burden, this method also achieves finite-time average consensus for any horizon value, provided that the algorithm runs for sufficiently large number of observations, and each node computes less conservative set-valued estimates. The result is relevant in practice to determine a stopping time in a faulty environment, which is not a straightforward issue due to the iterative nature and uncertainty generated by the random choice of communicating nodes. If conditions for finite-time convergence are not met within the time that the algorithm is running, asymptotic convergence of the state of the nodes is also provided.

We envisage as directions for future work, the study of additional properties of specific classes of algorithms. In particular, structural premises that allow to eliminate certain sequences of matrices  $A(k)$  which are irrelevant for the computation of the SVO. In essence, associated with the results presented in this chapter, such a mechanism would decrease the complexity even further and broaden the spectrum of application of the proposed fault detection. Another line of possible research would be to integrate the SVO in a fault isolation mechanism as to progress towards a fault correction scheme where the nodes would, after detecting a fault, isolate the faulty node and correct the state of the algorithm to a value closer to the true state if there was no fault. Such a goal poses very interesting research problems.



## SVOs FOR LPV SYSTEMS WITH COPRIME FACTORIZATION

### 5.1 Introduction

The problem of detecting faults in the context of Linear Parameter-Varying (LPV) systems relates to that of determining if the current observations of the *true* system are compatible with the theoretical fault-free model. In particular, the framework of LPV systems is considered in this chapter since applications of fault detection mechanisms for LPV systems are commonly found in industrial processes (see examples in the survey in [RS00]). In addition, the distributed algorithms presented in the Chapter 2 and Chapter 3 can also be viewed as LPV systems driven by dynamics dependent on stochastic or deterministic actions that can be measured only at the current time instant.

The study of fault detection problems has been a long standing research topic, since the early 70's (see [Wil76]), but still poses remarkable challenges to both the scientific community and the industry (see, for example, the survey in [HKKS10] and the references therein). Classical fault detection methods such as the ones proposed in [Wil76], [Bar07], [BB04], [Duc09], [MGB05], [DF90] and [NVR08], rely on designing filters that generate residuals that should be *large* under faulty conditions. These strategies aim to derive bounds (or thresholds) on these residuals that can be used to decide whether a fault has occurred or not. However, calculation of these thresholds is typically cumbersome or poses stringent assumptions on the exogenous disturbances and measurement noise acting upon the system. Many implementations of residual-based Fault Detection and Isolation (FDI) techniques are available in the literature such as [HKY98], [Sau05], [CP12] and [Duc15].

In [RSSA10], [RS13], the authors develop the idea of using Set-Valued Observers (SVOs), whose concept was introduced in [Wit68] and [Sch68] (further information can be found in [Sch73] and [MV91] and the references therein) for fault detection by resorting to a model invalidation (falsification) approach. The method is particularly interesting in the sense that it is able to handle a relatively large class of dynamic models, while also reducing the conservatism

of the results by incorporating the information of past observations in the construction of the current set-valued state estimates. However, two main drawbacks of the approach can be identified: the convergence properties are shown for stable systems only, and the calculation of the set-valued state estimates requires a significant computational effort. The latter limitation is a consequence of the need to increase the horizon of the observations to produce accurate results. The aim of this chapter is to extend the SVO-based fault detection method in order to cope with unstable systems and to reduce the necessary horizon value for the class of LPV systems. It is a generalization of existing results incorporating a left-coprime factorization into the design of SVOs for Linear Time-Invariant systems (LTI) [RSA14].

Related to the problem of fault detection is how to distinguish between two different faults in the system assuming they are *distinguishable* in some sense, as formally defined in the sequel. The state-of-the-art methods for fault isolation resorting to the concept of model invalidation using SVOs are based either on designing a filter for each fault, as in [RS13] and [CRT<sup>+</sup>15], or on storing the observations and running a constant number of SVOs several times, where each model considers a subset of the fault signals, as in [BRSO15]. Both methods yield significant limitations: the former requires an exponential number of SVOs with the number of faults to be considered, if no assumptions are made on the maximum number of concurrently occurring faults; and the latter, although reducing the required computational cost by only running the fault isolation filters on a subset of the fault space, still poses constraints on the applicability for time-sensitive applications and prevents possible parallelization of the computations since the new subset of the fault signal to be considered might depend on the result of running the SVOs on previous partitions.

### 5.2 Main Contributions and Organization

This chapter starts by reviewing all the necessary tools found in the literature that are needed to use the coprime factorization to achieve convergence results meaningful to reduce the aforementioned problems. It then progresses to give a different perspective on how the equations for the SVOs can be used to perform fault isolation without having an exponential increase in the number of filters.

The advantages, presented in the paper [SRHS17a], can be summarized in four topics:

- The use of a left coprime factorization for LPV systems enables SVO-based fault detection, even when the plant is unstable;
- The convergence proof of the method is provided for a broad class of LPV systems and for any horizon greater than  $n_x$ , the size of the state space, by exploiting the properties of deadbeat observers;
- Fault isolation is addressed by including the fault signal into the model of a single SVO and retrieving it through a projection, which reduces the amount of required computations

especially in the case of a large number of faults and no bounds on the number of concurrent faults;

- The computation of the set-valued estimates of the fault signal allows the incorporation of linear constraints involving the fault signal that are common, for instance, when dealing with budget constraints for an attacker in a network.

### 5.3 Problem Statement

We consider the dynamics of a non-faulty system, described by a Linear Parameter-Varying (LPV) model of the form:

$$\begin{cases} x(k+1) = A(\rho(k))x(k) + B(\rho(k))u(k) + L(\rho(k))d(k) \\ y(k) = C(\rho(k))x(k) + v(k) \end{cases} \quad (5.1)$$

with bounded unknown exogenous disturbances,  $d(\cdot) \in \mathbb{R}^{n_d}$ , bounded unknown sensor noise,  $v(\cdot) \in \mathbb{R}^{n_v}$ , uncertain initial state  $x(0) \in X(0)$ , where  $X(0)$  is a set that is guaranteed to contain the initial state  $x(0)$ . Matrices  $A(\rho(k))$ ,  $B(\rho(k))$ ,  $L(\rho(k))$ , and  $C(\rho(k))$  are parameter-dependent, and  $\rho(k)$  is assumed to be measured. The state is described by  $x(k) \in \mathbb{R}^{n_x}$  and the known input signal by  $u(k) \in \mathbb{R}^{n_u}$ . Without loss of generality, it is assumed that  $|d_i(k)| \leq 1, \forall k \geq 0, 1 \leq i \leq n_d$  and  $|v_i(k)| \leq v^*, \forall k \geq 0, 1 \leq i \leq n_v$ . To lighten the notation, the dependence on the parameter  $\rho$  will be omitted, when clear from context. As an example, we will write  $A_k$  to denote  $A(\rho(k))$ , whenever the parameter-dependence can be inferred from context.

**Problem 1** (Fault Detection). *The problem of fault detection relies on a model invalidation approach. In that sense, all types of faults that can be detected generate output sequences of the true system,  $y(k)$ , for which do not exist initial conditions  $x(0)$ , disturbances  $d(k)$ , noise signals  $v(k)$ , and values of the parameter  $\rho(k)$ , such that the output can be generated by model (5.1).*

We require the following definition to state the main assumption of this chapter.

**Definition 5.1** (Uniformly  $n_x$ -step Observable [Lev96]). *A system (5.1) is said to be uniformly  $n_x$ -step observable if the observability matrix*

$$O(k, k+n_x) := \begin{bmatrix} C_k \\ C_{k+1}\Phi(k+1, k) \\ \vdots \\ C_{k+n_x-2}\Phi(k+n_x-2, k) \\ C_{k+n_x-1}\Phi(k+n_x-1, k) \end{bmatrix}$$

*has rank equal to  $n_x$  for any parameter value  $\rho(k)$ , where*

$$\Phi(k, k_0) := \begin{cases} I & k = k_0 \\ A_{k-1}A_{k-2} \cdots A_{k_0+1}A_{k_0} & k > k_0 \end{cases}.$$

The main assumption throughout this chapter is summarized in Assumption 5.1.

**Assumption 5.1.** *The system described by (5.1) is uniformly  $n_x$ -step observable as in Definition 5.1.*

Within the scope of fault detection, it may be required to maintain a set of all possible state realizations at each time instant to determine if the observations are consistent with the fault-free model in (5.1). We resort to Set-Valued Observers (SVOs) specified in the previous chapter but adapting to the more general model of this chapter. The notation  $\bar{Z} := \begin{bmatrix} Z \\ -Z \end{bmatrix}$ , for a matrix  $Z$ , and  $\bar{v} := \begin{bmatrix} v \\ -v \end{bmatrix}$ , for a vector  $v$  will be used to shorten the following equations. Considering the information of a single measurement (i.e., by setting the horizon  $N = 1$ ),  $X(k+1)$  can be described as the set of points,  $\mathbf{x}$ , satisfying

$$\underbrace{\begin{bmatrix} M(k)A_k^{-1} & -M(k)A_k^{-1}L_k \\ \bar{C}_{k+1} & 0 \\ 0 & \bar{I} \end{bmatrix}}_{M(k+1)} \begin{bmatrix} \mathbf{x} \\ \mathbf{d} \end{bmatrix} \leq \underbrace{\begin{bmatrix} m(k) + \bar{u}(k, 1) \\ \bar{y}(k+1) + v^*1 \\ 1 \end{bmatrix}}_{m(k+1)}, \text{ for some } \mathbf{d} \quad (5.2)$$

where we used the notation  $\bar{u}(k, N) := \sum_{\tau=1}^N M(k)\Phi(k+1, k-\tau+1)^{-1}B(k)u(k-\tau+1)$ . This procedure assumes an invertible matrix of the dynamics,  $A_k$ , at each time instant. When this is not the case, we can adopt the strategy in [ST99] and solve the inequality

$$\begin{bmatrix} \bar{I} & -\bar{A}_k & -\bar{L}_k \\ 0 & 0 & \bar{I} \\ \bar{C}_{k+1} & 0 & 0 \\ 0 & M(k) & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{x}^- \\ \mathbf{d} \end{bmatrix} \leq \begin{bmatrix} \bar{B}_k u(k) \\ 1 \\ \bar{y}(k+1) + v^*1 \\ m(k) \end{bmatrix}. \quad (5.3)$$

By applying the Fourier-Motzkin elimination method [KG87] to remove the dependence on  $\mathbf{x}^-$ , we obtain the set described by  $M(k+1)\mathbf{x} \leq m(k+1)$ .

The above computations assume a horizon value  $N = 1$ , i.e., only the measurements from time  $k$  and the input signal from time  $k-1$  are used to compute the set-valued estimate of the state at time  $k$ . Due to the uncertainty in the initial state or the use of an approximation,  $\tilde{X}(k)$ , to set  $X(k)$  (for example, to avoid the number of vertices of the polytope to render the calculation of the Fourier-Motzkin elimination method intractable), one might consider including past measurements to improve detection, at the expenses of a higher computational cost, by extending the previous inequalities to a general horizon  $N$ . In doing so, it may reduce the conservatism of the set-valued state estimate, as shown in [RSA14]. Let us introduce the notation  $M_N(k+1)$  to represent the construction of matrix  $M(k+1)$ , in the definition of set  $X(k+1)$ , for a given horizon  $N$ . If  $A_k$  is non-singular, then the following inequality holds

$$\left[ \begin{array}{c} M_N(k+1) \\ 0_{(N+2n_d) \times n_x} \quad I_N \otimes \bar{I}_{n_d} \end{array} \right] \left[ \begin{array}{c} \mathbf{x} \\ \mathbf{d}(k) \\ \vdots \\ \mathbf{d}(k-N+1) \end{array} \right] \leq \underbrace{\left[ \begin{array}{c} m(k) + \tilde{u}(k, 1) \\ \bar{y}(k+1) - \nu^* \mathbf{1} \\ \vdots \\ m(k-N+1) + \tilde{u}(k, N) \\ \bar{y}(k-N+2) - \nu^* \mathbf{1} \\ 1 \\ \vdots \\ 1 \end{array} \right]}_{m(k+1)}. \quad (5.4)$$

for some possible values of  $\mathbf{d}(k), \dots, \mathbf{d}(k-N+1)$  and where  $M_N(k+1)$  is defined by

$$M_N(k+1) := \left[ \begin{array}{cccc} M_{N-1}(k+1) & 0 & & \\ M(\eta)\Phi(k+1, \eta)^{-1} & -M(\eta)\Phi(k+1, \eta)^{-1}L_k & \cdots & -M(\eta)\Phi(\eta+1, \eta)^{-1}L_\eta \\ C_{\eta+1}\Phi(k+1, \eta+1) & 0 & \cdots & 0 \end{array} \right]$$

where  $\eta = k - N + 1$ . For the sake of completeness, if  $A_k$  is non-invertible, then the following alternative inequality will hold

$$M_N(k+1) \left[ \begin{array}{c} \mathbf{x}(k+1) \\ \mathbf{x}(k) \\ \mathbf{d}(k) \\ \mathbf{x}(k-1) \\ \mathbf{d}(k-1) \\ \vdots \\ \mathbf{x}(k-N+1) \\ \mathbf{d}(k-N+1) \end{array} \right] \leq m_N(k+1)$$

where

$$M_N(k+1) := \left[ \begin{array}{ccccc|cc} M_{N-1}(k+1) & & & & & 0 & \\ \bar{I} & 0 & \cdots & 0 & -\bar{L}_k & -\bar{\Phi}(k+1, k-N+1) & -\bar{\Phi}(k+1, k-N+2)L_\eta \\ 0 & 0 & \cdots & 0 & 0 & 0 & \bar{I} \\ 0 & \bar{C}_{k-N+1} & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & M(k-N+1) & 0 \end{array} \right]$$

and

$$m_N(k+1) := \left[ \begin{array}{c} m_{N-1}(k+1) \\ \sum_{\tau=0}^{N-1} \bar{A}_k^\tau B_k u(k-\tau) \\ 1 \\ \bar{y}(\eta) + \nu^* \mathbf{1} \\ m(k-N+1) \end{array} \right].$$

In the next sections, we review the design of deadbeat observers and coprime factors which can be used together to achieve interesting properties for the SVOs.

## 5.4 Deadbeat Observers for LPV systems

In this section, we describe briefly, for the sake of completeness, the procedures found in the literature to design deadbeat observers for LPV systems satisfying Assumption 5.1. The existence of a deadbeat observer will be useful when proving the main result in this chapter in terms of the boundedness of the hyper-volume of the proposed SVO estimates.

In [Hos82], the author introduces a procedure to find a deadbeat observer for LPV systems with  $C_k$  to be a vector, represented by  $c_k$ . In the sequel, we recover this procedure, which can be extended for the case of a matrix  $C_k$  in a straightforward manner, by considering the right matrix division whenever a division involves vectors or matrices, as described next. The related observer dynamic system is described by the state  $z(k)$  with dynamics

$$z(k+1) = \Psi_k z(k) + B_k u(k) + K_k y(k) \quad (5.5)$$

with

$$\Psi_k = A_k - K_k C_k. \quad (5.6)$$

The estimation error is then given by

$$x(k) - z(k) = \Psi_{k-1} \Psi_{k-2} \cdots \Psi_0 (x(0) - z(0))$$

leading to the conclusion that a deadbeat observer must satisfy

$$\Psi_{k-1} \Psi_{k-2} \cdots \Psi_0 = 0. \quad (5.7)$$

A simple sequential algorithm will work for the scalar case when we have a vector  $c_k$  by solving

$$\Psi_{k-1} \Psi_{k-2} \cdots \Psi_0 \mathbf{e}_i = 0, \quad i = 1, 2, \dots, n_x$$

which is equivalent to (5.7). The approach proposed in [Hos82] is to solve

$$\begin{aligned} \Psi_0 \mathbf{e}_1 &= 0 \\ \Psi_1 \Psi_0 \mathbf{e}_2 &= 0 \\ &\vdots \\ \Psi_{k-1} \cdots \Psi_1 \Psi_0 \mathbf{e}_{n_x} &= 0 \end{aligned}$$

which imply that the deadbeat observer gain matrix  $K_k$  can be found using

$$K_k = \frac{A_k \Psi_{k-1} \Psi_{k-2} \cdots \Psi_{k-n_x+1} \mathbf{e}_i}{C_k \Psi_{k-1} \Psi_{k-2} \cdots \Psi_{k-n_x+1} \mathbf{e}_i} \quad (5.8)$$

where  $i = \min(k, n_x)$  and

$$\Psi_{-1} = \Psi_{-2} = \cdots \Psi_{-n_x+1} = I.$$

In order to extend the above calculations for the matrix case (i.e., when more than one measurement is available) one can use

$$K_k = A_k \Psi_{k-1}^{n_x-1} \mathbf{e}_k (C_k \Psi_{k-1}^{n_x-1} \mathbf{e}_k)^\dagger$$



where for a matrix  $Z$ , the notation  $Z^\dagger$  represents the Moore-Penrose pseudoinverse and with  $\Psi_{k-1}^{n_x-1} := \Psi_{k-1} \Psi_{k-2} \cdots \Psi_{k-n_x+1}$ .

Therefore, the computation of the deadbeat observer follows these steps

- Calculate the observer gain  $K_k$  using (5.8);
- Compute the next observer state transition matrix using (5.6);
- Update the observer state estimate via (5.5).

## 5.5 Coprime Factorization

A key tool to our method is the concept of coprime factorization for LPV systems, which allows, under certain assumptions stated in the sequel, to describe a dynamic system by means of the interconnection of two systems that are separately stable. For each of the subsystems, an SVO can be designed with guarantees of convergence of the set-valued state estimates. By this convergence, it is understood that the set-valued state estimates remain bounded, for bounded input and output plant signals, as formally presented next.

We start by introducing the definition of coprime factorization.

**Definition 5.2** (coprime factorizations [RPK92]). *A normalized left-coprime (respectively, right-coprime) factorization of an observable system  $P$  (satisfying Assumption 5.1) described by (5.1), defined by  $S_Q$  and  $S_G$  (respectively,  $\tilde{S}_Q$  and  $\tilde{S}_G$ ) is such that  $P = S_G^{-1} S_Q$  and  $S_Q X + S_G Y = I$  for some  $X, Y$  (respectively,  $P = \tilde{S}_Q \tilde{S}_G^{-1}$  and  $X \tilde{S}_Q + Y \tilde{S}_G = I$ ).*

In [FD07], a right-coprime factorization is given for nonstationary LPV systems and the corresponding factorization for stationary LPVs can be found in [BB97], [Bec06]. Similarly, we can obtain the left-coprime factorization for an observable system, such as in Definition 5.1,  $P = S_G^{-1} S_Q$ , which is given by

$$S_Q = \begin{bmatrix} A_k - K_k C_k & B_k - K_k D_k \\ R_k C_k & R_k D_k \end{bmatrix}, S_G = \begin{bmatrix} A_k - K_k C_k & -K_k \\ R_k C_k & R_k \end{bmatrix} \quad (5.9)$$

where  $K_k$  is such that  $A_k - K_k C_k$  is stable. Notice that such a matrix  $K$  is guaranteed to exist, due to the assumption of (5.1) being observable. In addition,  $R_k$  is non-singular.

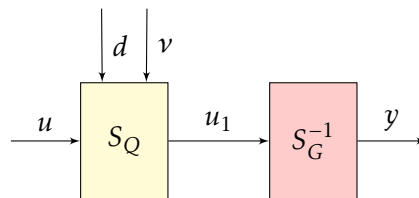


Figure 5.1: Schematic representation of the two coprime systems.

Figure 5.1 depicts the decomposition of the system obtained using the coprime factorization in (5.9) and stacking the exogenous inputs  $d$  and  $v$  in vector  $u$ . The two colors indicate the

separate parts that form each of the two subsystems. In this approach, the SVOs can be applied to each of the individual subsystems, since they are, by construction, stable. The stability condition was required in the proof of convergence (see [RS14]), since, intuitively, a sufficiently large horizon needs to be considered, so that the system dynamics over the horizon results into a contraction operator.

## 5.6 Fault Detection

In the previous sections, we introduced the building blocks to address the two main issues regarding the detection of faults using SVOs: the need for a large horizon value (see [RSA14]), and the assumption on the stability of the system (see [RS14]). These two problems are related to each other in the sense that, to ensure convergence, one would need to guarantee that the SVO, seen as an operator, is a contraction, for a sufficiently large horizon. This condition requires the system to be stable, and even in this case, can result in a potentially large horizon, which, in general, renders the problem computationally heavy.

The main idea behind this novel fault detection method revolves around the approach introduced in [RSA14] for LTI systems, which consists in applying the coprime factorization to the original system, thus yielding two stable subsystems - one that take the exogenous signals  $u$ ,  $v$  and  $d$ ; and another one that uses  $y$  - with both of them producing an internal variable  $u_1$ . The detection is performed by requiring the intersection of the set-valued estimates of the two SVOs for  $u_1$  to be non-empty, as described in the following proposition. An illustration of this procedure is depicted in Figure 5.2.

**Proposition 5.1.** *Consider an LPV system with dynamics given by (5.1), a coprime factorization given by (5.9), and sets  $X^{S_Q}(k)$  and  $X^{S_G}(k)$  respectively produced by the SVOs for the output of each of the subsystems  $S_Q$  and  $S_G$  in (5.9). A fault is detected at time instant  $k$  if  $X^{S_Q}(k) \cap X^{S_G}(k) = \emptyset$ .*

*Proof.* Let us prove by contradiction and therefore assume that  $X^{S_Q}(k) \cap X^{S_G}(k) = \emptyset$  and there is no fault. No fault means that an SVO returning the set  $X^P(k)$  for the output of the original system  $P$  in (5.1) satisfies

$$\forall k \geq 0 : y(k) \in X^P(k). \quad (5.10)$$

Having  $X^{S_Q}(k) \cap X^{S_G}(k) = \emptyset$  means that

$$\nexists u_1(k) : u_1(k) \in X^{S_Q}(k) \wedge u_1(k) \in X^{S_G}(k). \quad (5.11)$$

Combining (5.10) and (5.11), we get that  $P \neq S_G^{-1}S_Q$  since for system  $P$  there exists possible values for the initial conditions  $x(0)$  and signals  $u(\cdot)$ ,  $d(\cdot)$  and  $v(\cdot)$  that return all the outputs  $y(k)$  but the same is not true for the system  $S_G^{-1}S_Q$ . Thus, we reach a contradiction as we assumed systems  $S_Q$  and  $S_G$  were given as in (5.9).  $\square$

Proposition 5.1 motivates the introduction of the fault detection approach in Algorithm 4.

Another interesting issue is how to bound the horizon by a small value, since the computational burden grows exponentially with this variable. The concept of deadbeat observers plays a key role in providing such a result. Intuitively, it means that if the original system admits a deadbeat observer, then the associated set-valued state estimate can be bounded, since the term associated with the size of the previous estimate vanishes after  $n_x$  measurements, where  $n_x$  is the number of states in the system. This is one of the main results of this chapter and will be described next.

---

**Algorithm 4** Fault Detection of LPV systems using SVOs for a Coprime Factorization

---

**Require:** Set  $X(0)$  and an overbound for  $v(\cdot)$ .

**Ensure:** Fault detection, using SVOs with horizon equal to  $n_x$ .

```

1:  $G = \text{Compute\_deadbeat}()$  using (5.8)
2:  $\text{Factorize}()$  to obtain (5.9)
3:  $\text{Init\_SVO}_{S_Q}()$  using subsystem  $S_Q$ 
4:  $\text{Init\_SVO}_{S_G}()$  using subsystem  $S_G$ 
5:
6: for each  $k$  do
7:   /* Finding the set-valued estimates */
8:    $X^{S_Q}(k+1) = \text{update\_SVO}_{S_Q}(X^{S_Q}(k))$  using (5.4) with horizon =  $n_x$ 
9:    $X^{S_G}(k+1) = \text{update\_SVO}_{S_G}(X^{S_G}(k))$  using (5.4) with horizon =  $n_x$ 
10:
11:   /* Check if  $X^{S_Q}(k+1) \cap X^{S_G}(k+1)$  is empty */
12:   if  $X^{S_Q}(k+1) \cap X^{S_G}(k+1) = \emptyset$  then
13:     return System is faulty
14:   end if
15: end for

```

---

Algorithm 4 shows the pseudo-code for the fault detection described in this chapter. Notice that the maximum horizon needed is equal to the number of states based on the single assumption of the original system being observable in the sense of Definition 5.1. The described SVO-related functions can be implemented using the tools provided in [CRS15].

For Algorithm 4, it is possible to provide a result ensuring that the polytopic set-valued estimates do not grow unbounded both in the number of vertices and hypervolume. The set-valued estimates being bounded means that there exists an overbound set denoted by  $\hat{\Theta}(k)$ , after a number  $n_o$  of iterations satisfying  $n_o \geq n_x$ , that is bounded. During implementation, the polytopic sets are enlarged with a quantity equal to the maximal numeric error due to floating point limited precision of the machines to avoid false positives (see [CRT<sup>+</sup>15] for more details on this issue). We make the following assumption that the maximal numeric error  $\epsilon(k)$  of an SVO satisfies  $\epsilon(k) \leq \epsilon^* |x(k)|$ , for some  $0 \leq \epsilon^* < 1, \forall x(k) \in X(k)$ . In taking into account the numeric error, the result can be applied in practice and does not stand as a purely theoretical one.

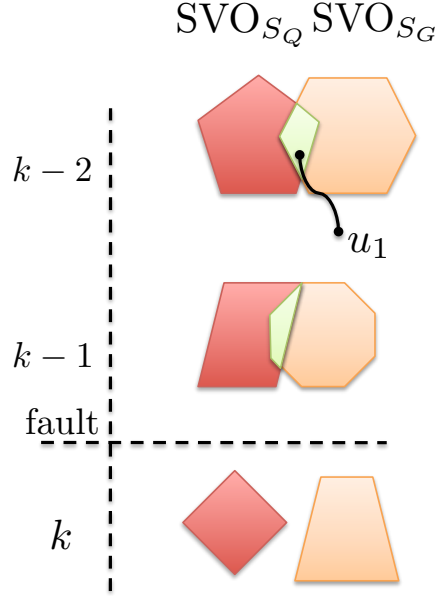


Figure 5.2: Illustration of the fault detection mechanism resorting to the intersection of the sets generated by the SVOs of each subsystem resulting from the coprime factorization.

**Theorem 5.1.** Consider an observable system described as in (5.1) with state  $x(k) \in \mathbb{R}^{n_x}$ , actuated by control input  $u(k) \leq u^* < \infty$ , with exogenous disturbances  $d(k)$  and with measurements  $y(k) \leq y^* < \infty$ , corrupted by additive noise  $n(k)$ , such that  $|d(\cdot)| \leq 1$  and  $|v(\cdot)| \leq v^*$ . Then, there exists a coprime factorization for (5.1) and the set-valued estimates produced by Algorithm 4 are bounded for any  $n_o \geq n_x$ .

*Proof.* The existence of a deadbeat observer comes directly from the assumption that the system is observable (see [Hos82]).

From the existence of a deadbeat observer, it is clear that

$$\forall k \geq 0 : \varphi(k + n_x, k) = 0 \quad (5.12)$$

holds for a choice of matrix sequence  $G_k$  computed using (5.8), where

$$\varphi(k, k_0) := \begin{cases} I_{n_x}, & \text{if } k = k_0 \\ (A_{k-1} - G_{k-1}C_{k-1}) \dots (A_{k_0} - G_{k_0}C_{k_0}), & \text{if } k > k_0 \end{cases}.$$

One needs to prove that SVO for system  $S_Q$ , and SVO for system  $S_G$ , produce bounded sets. The following result focus on SVO for system  $S_G$  and, for that reason, we drop superscript  $S_G$ .

Consider the smallest hypercubes,  $\hat{\Theta}(k), \hat{\Theta}(k+1), \dots, \hat{\Theta}(k+n_o)$  that contain the sets  $\hat{X}(k), \hat{X}(k+1), \dots, \hat{X}(k+n_o)$ , which represent the original set-valued state estimates  $X(k), X(k+1), \dots, X(k+n_o)$  plus the maximal numeric error  $\epsilon(k)$  at each time instant, satisfying the assumption stated before. For any  $n_o \geq n_x$ , an overly conservative estimate can be generated using the inequality

$$|x(k + n_o)| \leq |\varphi(k + n_o, k)x(k)| + \epsilon^*|x(k)| + \delta_{n_o} \quad (5.13)$$

where

$$\delta_{n_o} = \max_{y(k), \dots, y(k+n_o-1)} \left| \sum_{\tau=k}^{k+n_o-1} [\varphi(k+n_o, \tau+1) G_{\tau} y(\tau)] \right|.$$

From the deadbeat condition (5.12), we get that the expression (5.13) simplifies to

$$|x(k+n_o)| \leq \epsilon^{\star} |x(k)| + \delta_{n_o}.$$

However, given that by assumption  $\epsilon^{\star} < 1$  and  $|y(k)| \leq y^{\star} < \infty$ , there exists  $\delta^{\star}$  such that  $|\delta_{n_o}| \leq \delta^{\star} < \infty$  thus proving the boundedness of SVO  $S_G$ . A similar result can be found for SVO  $S_Q$ , which concludes the proof.  $\square$

## 5.7 Fault Isolation

In this section, we show how the SVOs can be employed in fault isolation resorting to estimating the fault signal instead of the concept of model invalidation. In doing so, only one SVO is required for fault isolation instead of using a combinatorial number corresponding to each combination of faults.

The model in (5.1) used for fault detection is now extended by considering the fault signal as an external component added to the state dynamics as

$$\begin{cases} x(k+1) = A(\rho(k))x(k) + B(\rho(k))u(k) + L(\rho(k))d(k) + F(\rho(k))f(k) \\ y(k) = C(\rho(k))x(k) + v(k) \end{cases} \quad (5.14)$$

where matrix  $F(\rho(k))$  is known at each time instant and determines what are the possible states the fault signal can corrupt. It is stressed that, from a physical perspective, these can be interpreted as actuator faults. The dynamics in (5.14) can be rewritten to match (5.1) as follows:

$$\begin{aligned} x(k+1) &= A(\rho(k))x(k) + B(\rho(k))u(k) + \begin{bmatrix} L(\rho(k)) & F(\rho(k)) \end{bmatrix} \begin{bmatrix} d(k) \\ f(k) \end{bmatrix} \\ y(k) &= C(\rho(k))x(k) + v(k) \end{aligned} \quad (5.15)$$

Unlike the work described in [RS13], [CRT<sup>+</sup>15], and [BRSO15], the strategy proposed in this section is based on inverting the logic applied to fault detection. I.e., whereas in the previous algorithms, SVOs were used to produce set-valued estimates of the state compliant with the system dynamics, bounds for the initial state, disturbances, and noise signals, the approach proposed in this section is to estimate the fault signal itself. In the previous view of the problem, when the set for the state estimates is empty, the measurements cannot be generated by the model, and thus a fault is detected. In the novel approach proposed herein, the fault input signal  $f(\cdot)$  appears as variable in the definition of the polytope  $\tilde{X}(k)$ . By means of a projection onto those coordinates, set-valued estimates of the fault signals are obtained. The SVO will produce a set for the combinations of possible faults given the measurements, dynamics and bounds for the system.

## Chapter 5: Coprime Factorization

Equation (5.15) results in a different set of equations for the SVO definition. Only the most generic case, where  $A$  is singular is presented. The calculations when  $A$  is nonsingular may be obtained in a straightforward manner by following the same procedure. In addition, it is also considered that the faults may also satisfy certain constraints. For instance, let us consider that we are playing against an adversary which is changing some of the state variables in a power network or in a budget-constrained scenario. He or she will have a finite amount of power or budget to compromise the system. Such an example motivates the introduction of a linear constraint of the type  $Rf(k) \leq 1$ , for some known matrix  $R$ .

$$M_N(k+1) \begin{bmatrix} \mathbf{x}(k+1) \\ \mathbf{x}(k) \\ \mathbf{d}(k) \\ \mathbf{f}(k) \\ \mathbf{x}(k-1) \\ \mathbf{d}(k-1) \\ \mathbf{f}(k-1) \\ \vdots \\ \mathbf{x}(k-N+1) \\ \mathbf{d}(k-N+1) \\ \mathbf{f}(k-N+1) \end{bmatrix} \leq m_N(k+1) \quad (5.16)$$

where

$$M_N(k+1) := \left[ \begin{array}{c|ccc} M_{N-1}(k+1) & & & 0 \\ \hline \bar{I} & 0 & \cdots & 0 & \bar{L}_k & \bar{F}_k & -\bar{A}_k^N & -\bar{A}_k^{N-1}L_{k-N+1} & -\bar{A}_k^{H-1}F_{k-N+1} \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \bar{I} & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 & R \\ 0 & C_{k-N+1} & \cdots & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & M(k-N+1) & 0 & 0 \end{array} \right]$$

and

$$m_N(k+1) := \begin{bmatrix} m_{N-1}(k+1) \\ \sum_{\tau=0}^{N-1} \bar{A}_k^\tau B_k u(k-\tau) \\ 1 \\ 1 \\ \bar{y}(k-N+1) \\ m(k-N+1) \end{bmatrix}$$

with the base case for  $N = 1$  being

$$\begin{bmatrix} \bar{I} & -\bar{A}_k & -\bar{L}_k & -\bar{F}_k \\ 0 & 0 & \bar{I} & 0 \\ 0 & 0 & 0 & R \\ \bar{C}_k & 0 & 0 & 0 \\ 0 & M(k) & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{x}^- \\ \mathbf{d}(k) \\ \mathbf{f}(k) \end{bmatrix} \leq \begin{bmatrix} \bar{B}_k u(k) \\ 1 \\ 1 \\ \bar{y}(k+1) \\ m(k) \end{bmatrix}$$

In this new setting, the set  $\tilde{X}(k)$  must be projected onto the coordinates corresponding to  $\mathbf{f}(\cdot)$  to obtain a set where the fault signal belongs. Other restrictions can be introduced relating

the fault signal in different time instants, by including rows in the matrix  $M_N(k)$  corresponding to the known restrictions. The fault detection and isolation condition is presented in the next proposition that comes directly from the definition of SVOs.

**Proposition 5.2.** *Given a model (5.14) that includes a fault signal  $f(k)$ ,  $\forall k : |d(k)| \leq 1 \wedge |v(k)| \leq \bar{v}$  and  $X(0)$  such that  $x(0) \in X(0)$ , the following statements are true:*

- *A fault exists if  $0 \notin P_f \tilde{X}(k)$ , where  $P_f \tilde{X}(k)$  is the projection operator of the polytope  $\tilde{X}(k)$  onto the coordinates of  $f$  and  $\tilde{X}(k) := \text{Set}(M_N(k), m_N(k))$  as in (5.16);*
- *The  $i$ th fault is isolated from the remaining  $n_f - 1$  possible faults identified in the set  $S := \{e_{j_1}, \dots, e_{j_{n_f-1}}\}$  if  $\text{span}(S) \notin P_f \tilde{X}(k)$ , where the function  $\text{span}(S) := \{0 + \sum_{j \in S} \lambda_j e_j, \lambda_j \in \mathbb{R}\}$ .*

Proposition 5.2 translates the detection of a fault as the set obtained by projecting onto the coordinates of the fault signal to include the origin. By definition, if this is not the case, it does not exist initial conditions  $x(0) \in X(0)$ , disturbances signal  $d(k)$  and noise  $v(k)$  respecting the bounds and parameter  $\rho(k)$  such that the observations  $y(k)$  are produced without having a non-zero signal  $f(k)$ . Similarly, a fault is isolated if the produced observations  $y(k)$  can only be reproduced if the signal  $f(k)$  must coincide with one of the faults and not all the remaining.

## 5.8 Example and Simulations

In this section, an example is provided to illustrate how to compute the deadbeat observer gain and how the SVOs can be designed. The same example is used to depict the main features of the SVO-based fault detection using the coprime factorization approach.

In this chapter, we will consider an oscillator model with a mass of 10 kg connected to a spring, with its spring coefficient constant and equal to 1. The continuous time dynamic model can be described by the following system matrices

$$A_c(\rho(k)) = \begin{bmatrix} 0 & 1 \\ -\frac{1}{10} & -\frac{\rho(k)}{10} \end{bmatrix}, B_c = \begin{bmatrix} 0 \\ \frac{1}{10} \end{bmatrix}, C_c = \begin{bmatrix} 1 \\ 0 \end{bmatrix}^T, D_c = 0;$$

where parameter  $\rho$  is the damping coefficient and is assumed to be varying uniformly between 2.02 and 2.2 every 4 seconds. The system is discretized with a sampling period of 0.2 seconds, leading to a discrete-time system defined by the tuple of matrices  $(A(k), B, C, D)$ .

The first important aspect of the standard SVOs is the necessary horizon to ensure convergence. According to the results in [RS14], the product of matrices  $A(k)$  for the selected horizon need to satisfy the property that its singular values are less than 1. To guarantee that condition, in this example, one would need to set the horizon  $N = 42$ .

In this example, after the discretization, the dynamics matrix for the first two instants were computed to be

$$A_1 = \begin{bmatrix} 0.998 & 0.1959 \\ -0.0196 & 0.9585 \end{bmatrix}, A_2 = \begin{bmatrix} 0.998 & 0.1959 \\ -0.0196 & 0.9583 \end{bmatrix}$$

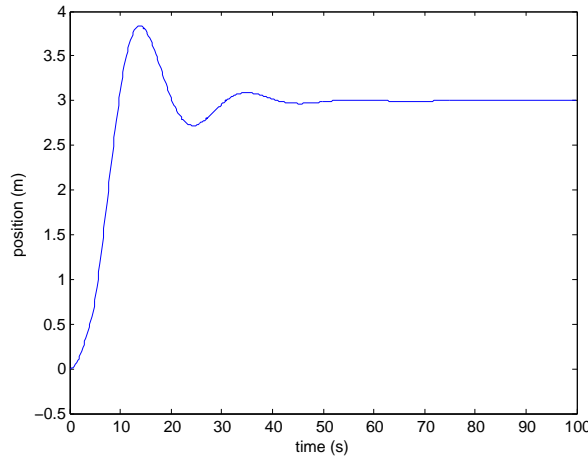


Figure 5.3: Output of the mass-spring-dashpot system with a fault introduced after 4 seconds.

which makes

$$K_1 = A_1 e_1, \Psi_1 = \begin{bmatrix} 0 & A_1 e_2 \end{bmatrix}, K_2 = \frac{A_2 A_1 e_2}{0.1959}$$

according to equations (5.6) and (5.8). The gain matrices  $K_k$  have the deadbeat property and were then used to compute the coprime factorization.

Firstly, we obtained the coprime factorization of the model

$$\begin{aligned} x(k+1) &= A_k x(k) + Bu(k) + Ld(k) \\ y(k) &= Cx(k) + v(k), \end{aligned}$$

where we assumed a matrix  $L = \begin{bmatrix} 2 & 1 \end{bmatrix}^T$ . The coprime factor  $S_Q$  is described by

$$\begin{aligned} x_Q(k+1) &= \Psi_k x_Q(k) + Bu(k) + Ld(k) - K_k v(k) \\ u_1(k) &= Cx_Q(k) + v(k) \end{aligned},$$

where the computation of  $\Psi(k)$  followed (5.6). Subsystem  $S_G$  is given by

$$\begin{aligned} x_G(k+1) &= \Psi_k x_G(k) - K_k y(k) \\ u_1(k) &= Cx_G(k) + y(k) \end{aligned}.$$

We start by depicting in Figure 5.3, the output of the system where a fault has been introduced after 4 seconds of the beginning of the simulation and detected by the SVO after 1 second (i.e., 5 sampling periods). We simulated faults translating a loss of actuation and for that purpose the true model of the plant is given by

$$\begin{cases} x(k+1) = A_k x(k) + B(u(k) + f(k)) + Ld(k) \\ y(k) = Cx(k) + v(k) \end{cases}$$

It is stressed that, even though in the presence of a fault, the output of the system does not exhibit any abnormal or easy-to-spot behavior. This motivates the need for automatic fault detection mechanisms, such as the SVO-based method presented in this chapter.



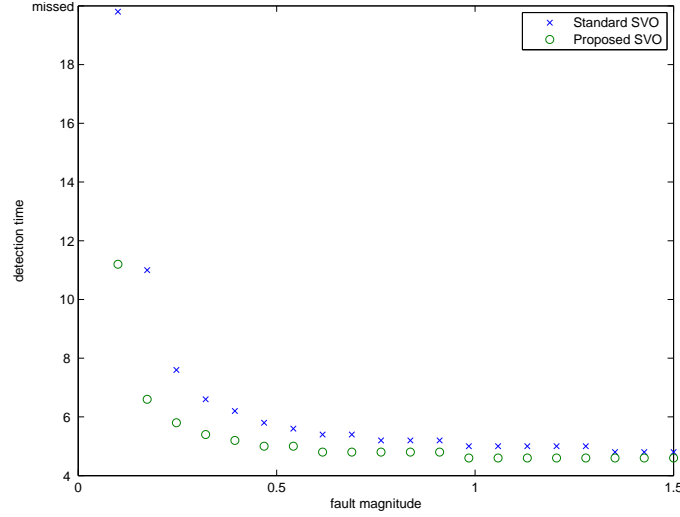


Figure 5.4: Detection time as a function of the magnitude of a constant fault introduced after 4 seconds.

In the simulations, we considered 4 different faults in order to assess the performance of the SVO-based method with coprime factorization in comparison with the standard fault detection mechanism [RS13]. To make the results comparable, we set both the standard and the coprime implementation with a horizon equal to 2 and resort to a hyper-parallelepiped overbound instead of the Fourier-Motzkin elimination method. This has significant improvements in the amount of time it is required to compute a single iteration of the algorithms.

In Figure 5.4, it is depicted the detection time instant for a constant fault injected after 4 seconds into the simulation. The faulty term is injected as an input with magnitude ranging from 0.1 to 1.5. However, after multiplying by matrix  $B$  it translates into a magnitude smaller than  $3 \times 10^{-2}$  for the 1.5 case and is comparable with the remaining faults simulated in this section.

Since it is deterministic, the constant fault illustrates a more predictable behavior and shows the decreasing trend between fault magnitude and the detection time. When the magnitude reaches 0.62, the detection time for the SVO with coprime factors is equal to 0.4 which corresponds to two discrete time steps (i.e., number of measurements required for detection equal to the number of states). Another interesting aspect is that, for the SVOs with coprime factors, the fault was detected even for small magnitudes whereas the error introduced by the overbound in the standard case prevented this detection.

The deterministic constant fault is a rather simplistic type of fault and does not stress the substantial difference and motivation to adopt the SVOs with the coprime factorization. In the aforementioned scenario, it amounted to a slower detection and required the magnitude of the fault to be higher to get a successful detection by the standard SVO. The second faulty case considers a model of the mass-spring-dashpot system corrupted by a random signal added to its state and drawn from a standard uniform distribution between zero and the maximum fault

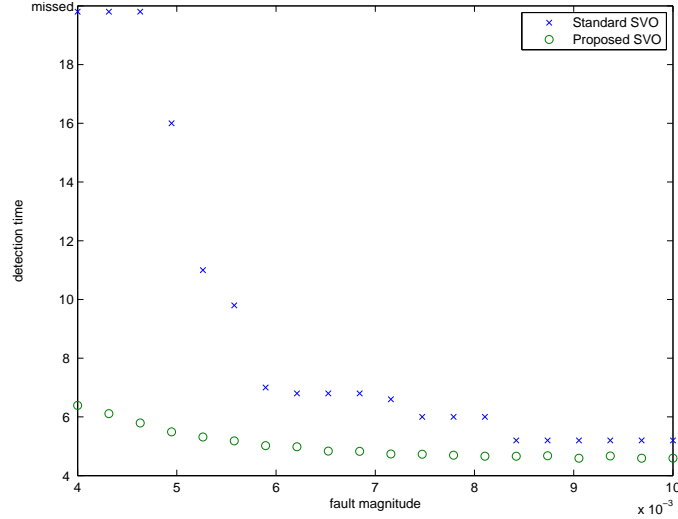


Figure 5.5: Mean detection time as a function of the magnitude of a random fault introduced after 4 seconds.

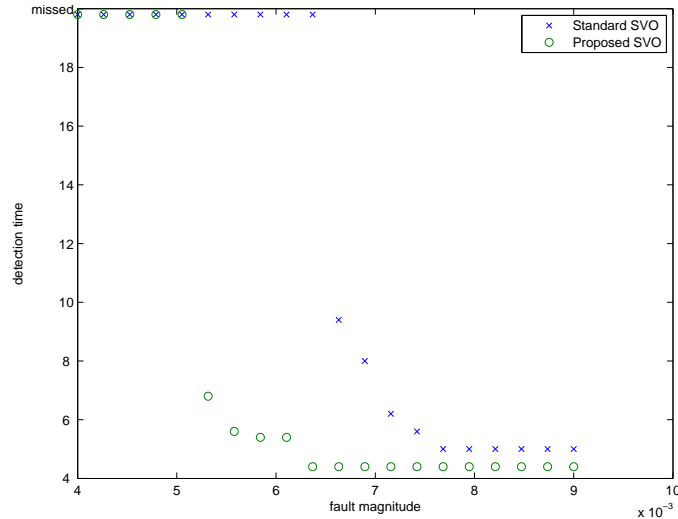


Figure 5.6: Detection time as a function of the magnitude of a sinusoid fault introduced after 4 seconds.

magnitude.

The mean detection times of a Monte Carlo simulation with 100 extractions for the random fault are shown in Figure 5.5. The smooth trend line for both the standard and the coprime SVO implementations is lost as it depends on the actual random sequence of the fault. The difference between mean detection times increases since the random fault is somehow more challenging, given that not only can the signal vary and be close to zero in some instants (which is almost fault-free), but also because the dynamics can cancel out current values with updated past values. Using the coprime-based method it is shown that, for some signals with small magnitude, the detection is possible whereas it is missed by the standard procedure.

The intuition gained with the two previous setups motivated the study of faults where the signal changes between positive and negative values. Such faults impact on the conservativeness

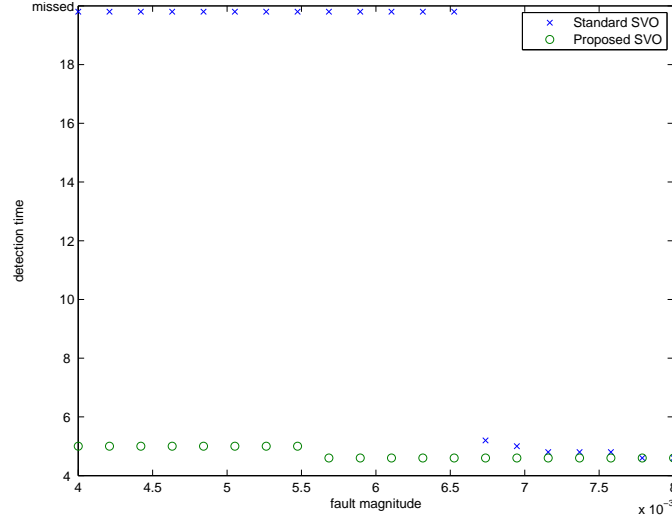


Figure 5.7: Detection time as a function of the magnitude of an alternating fault introduced after 4 seconds.

of the set-valued estimates and, as a consequence, on the detection procedure. To that extent, a sinusoidal fault was injected by simply adding a disturbance term equal to the magnitude of the fault multiplied by the sine wave.

In Figure 5.6, the results for the sinusoidal fault are presented. An interesting result is that, apart from a transient, the detection time either approaches the minimum time of 0.4s or there is a missed detection. For magnitudes between  $5 \times 10^{-3}$  and  $6.5 \times 10^{-3}$  only the coprime-based SVOs are able to perform the detection. For the remaining values of the fault, the standard SVOs perform poorly with detection times representing at least a two-fold increase.

The aforementioned considerations can be made more obvious by considering a fault that has constant absolute value but alternating between positive and negative every couple of discrete-time instants. The results are depicted in Figure 5.7, where the *binary* behavior of the standard technique is either the faults are detected in minimum time or not detected at all, while the proposed technique is able of detecting always the fault in 2 or 3 discrete time instants. A fault alternating in sign stresses how the conservatism of past iterations affects current set-valued estimates. In the standard case, it represents a major issue and detection happens for magnitudes greater than  $6.7 \times 10^{-3}$  whereas for small values as  $4 \times 10^{-3}$ , the coprime approach detects faults in a time close to the minimum.

The above simulations illustrated two key features of the proposed method, namely that conservative past estimates have a small impact in future iterations after a number of discrete-time instants equal to the size of the state space, and also that the proposed technique is suitable for the studied faults, achieving better detection times even for small magnitudes that would be missed by the standard procedure.

### 5.9 Conclusions

This chapter addressed the problem of designing Set-Valued Observers (SVOs) for Linear Parameter Varying (LPV) systems in the presence of noise and disturbances. Two main issues are of interest, namely, the need for large horizon values to ensure boundedness of the set-valued estimates; and allowing the SVOs to model unstable dynamics.

The solution adopted herein revolved around the concept of left-coprime factorization in order to design two subsystems that are stable, and to compute set-valued state estimates for each of the two subsystems. As a consequence, the dynamics of the subsystems can be used to construct a deadbeat observer gain matrix and use it with the SVOs, which reduces the necessary horizon to the number of states of the plant. It was shown that the set-valued estimates are bounded for the broad class of LPV systems.

The performance was evaluated by simulation to illustrate both the increase in speed of detection and the improvements in terms of better accuracy, since the aforementioned method reduces the conservatism of the final solution. Four different classes of faults were simulated for signals with the following characteristics: constant, random, sinusoidal, and constant amplitude but with changing signs.

The constant fault signal illustrated the decrease in detection time as the magnitude of the fault increases. The gap between the standard and the coprime approach increased as the magnitude of the fault decreases and follows a similar trend towards the minimum detection time. The constant signal represents an “easier” instance of the problem, as there is no cancellation of the current fault by the past values updated by the dynamics. This motivated simulating additional classes of faults such as those characterized by random signals.

Stochastic signals can represent, for example, unmodeled disturbances that need to be detected to avoid compromising the system. These are harder to detect in the sense that the fault signals can in some instants be close to zero (i.e., no fault) and then shift to a fault. The coprime-based SVOs achieved faster detection, at least in the example provided, even for small magnitudes of the signals.

Two other classes of faults were also simulated: a sinusoid signal and a constant amplitude with sign changing every two sampling times. In both cases, an interesting behavior emerged where either the fault was detected in a number of instants close to the minimum or was not detected at all. The coprime factorization-based approach allowed the detection of signals with much smaller magnitude, given that the conservatism of prior estimates is eliminated for a sufficiently large number of measurements. This contrasts with the standard procedure where these faults were harder to detect.

# 6

## FAULT DETECTION AND ISOLATION IN DETECTABLE SYSTEMS

### 6.1 Introduction

Performing fault detection in the context of cyber-physical systems can be difficult to address because the observability of the system can be affected. For example, having nodes with access to only local information or special network structures along with limited local state measurements can result in unobservable modes for the overall system. In this chapter, two types of cyber-physical systems are investigated: a group of dynamic physical systems cooperating over a network and smart power grids, where both attacks to the physical power grid infrastructure, as well as cyber attacks to the communication layer, can affect the overall network performance.

The motivation for this work is to provide tools to detect and isolate faults in cyber physical systems that have unobservable modes but are detectable. Current state-of-the-art techniques using set-valued estimators are not suitable for systems with unobservable modes and non-zero inputs as the disturbances and input signals increase the hypervolume of the set-valued estimates in each iteration, therefore resulting in divergent estimates.

The importance of addressing the fault detection (or state estimation) of a group of dynamic systems interconnected by a network is reported in [OSM04] and later in [MV09], where the detection is crucial given that a single malfunctioning node can severely impact on the overall network performance. Applications of such systems span the areas of mobile robots, cooperating unmanned vehicles tasks such as surveillance and reconnaissance, distributed state estimation, among others (see [ME14] and the references therein).

In the case of a smart grid, a network failure or malignant action can compromise its service which motivates the use of efficient fault detection mechanisms [ME10], [Ami11]. Besides failures and attacks to the physical power grid infrastructure, one must also consider cyber attacks to its communication infrastructure. Therefore, the problem of detecting faults and identifying where they are occurring in a network is considered in this chapter. To assess the performance of the techniques developed in the chapter, we adopt the linearized small signal version of the

structure-preserving model, composed by the linearized swing equation and the DC power flow equation. A comprehensive survey can be found in [FMXY12] regarding different aspects of the design of smart grids. The importance of this problem is reported in [ME10] and later in [Ami11].

There is a rich state-of-the-art for some specific problems regarding cyber-physical systems that resemble the model adopted in this chapter. In [ME14], one of the main results is showing that the overall system of a group of dynamic systems is unobservable when only considering relative information of the states. A transformation is introduced that allows to perform fault detection and isolation by considering the observable subspace of the overall system. The algorithm requires a centralized detection scheme. In this chapter, we derive an alternative approach based on Set-Valued Observers (SVOs), which enables a distributed detection for the observable subspace if we consider a strategy such that of [SRC<sup>+</sup>13] (described in Chapter 4).

In [SRC<sup>+</sup>13], the use of SVOs for distributed fault detection were firstly introduced for the specific case of consensus. The overall system is modeled as an Linear Parameter-Varying (LPV) system where communications are seen as a parameter-dependent dynamics matrix. Even though, the whole system is not observable in every time instant, for a sufficiently long time interval, the system is observable, as long as the underlying network topology is strongly connected. Whereas in [SRC<sup>+</sup>13], each node has access to its own state, and the state of one neighbor to which it communicates, in this chapter, it is assumed that nodes have access only to relative information. The distributed detection can also be improved by resorting to exchanging state estimates whenever the systems communicate or take measurements by using a similar algorithm to the one presented in [SRHS14].

An alternative method to the SVOs is the use of the reachability concept to construct set-valued estimates. The proposals in [ASB07] and [SC16] both resort to this concept and use zonotopes to define the sets where the state belongs. Zonotopes are a compromise of accuracy for performance in the sense that they are a subclass within polytopes. In addition, unions can be computed efficiently when compared to polytopes whereas intersections are much more efficient using polytopes. Our proposal focus on the use of polytopes since operations introduce less conservatism than zonotopes.

For the particular case of smart grids, other proposals have also been presented by the research community as alternative fault detection methods motivated by the increased interest for this topic by the industry. A survey focused on fault location methods for both transmission and distribution systems can be found in [Kez11].

In [MCHL14], faults are detected by constructing a  $\chi^2$ -detector that computes the  $\chi^2$  statistics of the residuals from a Kalman filter and compares them with the thresholds obtained from the standard distribution. Such a strategy is stochastic in nature and includes potential false-positives with a certain probability. The alternative approach presented in this chapter is deterministic and relies on a worst-case detection. A similar stochastic detection strategy can be

employed by using an extension of the framework proposed here, following the methodology described in [SRC<sup>+</sup>13].

Fault detection in smart grids has also been performed resorting to the concept of Petri Nets [CHPS11]. The procedure consists in mapping the possible concurrent actions of each of the nodes in the network to determine the current state of the system and checking if it is compatible with the measurements. In this chapter, we adopt a different methodology although the objective is the same, in the sense that we are computing a set of all possible states of the system.

In [GBG<sup>+</sup>11], the authors study the problem of undetectable faults due to the unobservable modes of the system. The fault detection is based on ensuring that the network is observable for a fixed number of compromised nodes by carefully selecting which states to measure. Although the focus is slightly different, the definition of the equation dictating the detection and isolation of faults is related. In [PDB11], one of the main results is to characterize detectability of faults both using dynamic and static procedures considering the dynamics of the network and no disturbances in the model.

In a different direction, [PBB11] and [PBB12] show that the theoretical condition for fault detectability and identifiability in the context of smart power grids is similar to that of detecting faults in consensus problems and amounts to studying the zero dynamics of the system given by the difference between the nominal “fault-free” and the one with the input fault signal. In this chapter, we rewrite the equations describing the set-valued estimates in a similar fashion, which describe *fast* SVO (fSVO) in the sense they are low-complexity methods by avoiding the need to resort to the Fourier-Motzkin elimination algorithm.

## 6.2 Main Contributions and Organization

The main contributions of this chapter, presented in the papers [SRHS15a] and [SRHS17b], are as follows:

- we show how to perform fault detection and isolation with SVOs for unobservable but detectable systems taking advantage of a coprime factorization;
- the incorporation of possible disturbances and sensor noise in the fault detection mechanism for smart grids;
- reformulation of the theoretical conditions for fault detection and isolation, which lead to a different set of SVO equations that when coupled together with a coprime factorization represent a more efficient method for fault detection without adding conservatism.

### 6.3 Observability issue

In this section, the problem of distributed fault detection addressed in this chapter is defined. Before introducing the model, we provide an overview of the abstract case of having  $\mathcal{S}$  dynamic systems interconnected by a bidirectional network topology. This introduces the observability issue that can arise in designing SVOs for fault detection in a distributed setting. We then focus on developing observers that are distributed and can deal with detectable systems.

#### 6.3.1 Systems of Systems

We analyze the problem described in [ME14], namely, a group of  $\mathcal{S}$  dynamic systems interacting according to a bidirectional network topology. The corresponding graph has  $\mathcal{S}$  vertices each representing an  $n$ -dimensional subsystem  $S_i$ , modeled as a Linear Time-Invariant (LTI) of the form:

$$S_i : \begin{cases} x_i(k+1) = Ax_i(k) + Bu_i(k) + Ff_i(k) + Ed_i(k) \\ y_{ij}(k) = C(x_i(k) - x_j(k)), j \in \mathcal{N}_i \end{cases}$$

where  $x_i \in \mathbb{R}^n$ ,  $u_i \in \mathbb{R}^{n_u}$ , represent the state and input signal of the  $i$ th subsystem. The unknown sequences  $f_i \in \mathbb{R}^q$  and  $d_i \in \mathbb{R}^r$  represent the fault and disturbance signals. Without loss of generality, it is assumed that  $|d_i(k)| \leq 1, \forall k \geq 0$ .

The fact that the dynamics matrices are equal for all of the subsystems complicates the problem as it renders the overall system unobservable.

The output of the  $i$ th system depends on all its neighbors  $j$ ,  $j \in \mathcal{N}_i$  :

$$y_i = \sum_{j \in \mathcal{N}_i} C(x_i - x_j)$$

which motivates the introduction of the graph laplacian matrix defined as

$$\mathcal{L}_{ii} = |\mathcal{N}_i|, \quad \mathcal{L}_{ij} = \begin{cases} -1, & \text{if } j \in \mathcal{N}_i \\ 0, & \text{if } j \notin \mathcal{N}_i \end{cases}$$

where  $|\mathcal{N}_i|$  is the number of neighbors of node  $i$ . By combining the state equations, the overall system is described by

$$\begin{aligned} x(k+1) &= \underbrace{(I_{\mathcal{S}} \otimes A)}_{A_{\mathcal{S}}} x(k) + \underbrace{(I_{\mathcal{S}} \otimes B)}_{B_{\mathcal{S}}} u(k) + \underbrace{(I_{\mathcal{S}} \otimes F)}_{F_{\mathcal{S}}} f(k) + \underbrace{(I_{\mathcal{S}} \otimes E)}_{E_{\mathcal{S}}} d(k) \\ y(k) &= \underbrace{(\mathcal{L} \otimes C)}_{C_{\mathcal{S}}} x(k) \end{aligned} \tag{6.1}$$

where  $x := [x_1^T \cdots x_{\mathcal{S}}^T]^T$  (i.e.,  $n_x = n\mathcal{S}$ ),  $u := [u_1^T \cdots u_{\mathcal{S}}^T]^T$ ,  $f := [f_1^T \cdots f_{\mathcal{S}}^T]^T$ ,  $d := [d_1^T \cdots d_{\mathcal{S}}^T]^T$  and  $y := [y_1^T \cdots y_{\mathcal{S}}^T]^T$ . As shown in Lemma 1 of [ME14], this system is always unobservable and a transformation is proposed to extract the observable subsystem in the following fashion.

Let

$$T := T_s^{-1} \otimes I_n$$



where

$$T_s^{-1} := \begin{bmatrix} 1 & 0_{S-1}^\top \\ -1_{S-1} & I_{S-1} \end{bmatrix}.$$

Using the transformation of state given by  $T$  such that  $x \rightarrow Tx$  yields an observable decomposition for the system due to the property of the Laplacian matrix

$$T_s^\top \mathcal{L} T_s = \begin{bmatrix} 0 & 0 \\ 0 & \mathcal{L}_r \end{bmatrix}$$

and the observable subsystem is now defined as

$$\begin{aligned} \bar{x}(k+1) &= \underbrace{(I_{S-1} \otimes A)}_{\bar{A}_S} \bar{x}(k) + \underbrace{(I_{S-1} \otimes B)}_{\bar{B}_S} \bar{u}(k) + \underbrace{(I_{S-1} \otimes F)}_{\bar{F}_S} \bar{f}(k) + \underbrace{(I_{S-1} \otimes E)}_{\bar{E}_S} \bar{d}(k) \\ \bar{y}(k) &= \underbrace{(\mathcal{L}_r \otimes C)}_{\bar{C}_S} \bar{x}(k) \end{aligned}$$

where  $\bar{x}_i := x_i - x_1$ ,  $\bar{u}_i := u_i - u_1$ ,  $\bar{f}_i := f_i - f_1$  and  $\bar{d}_i := d_i - d_1$  for  $2 \leq i \leq S$ .

The case in which the system is unobservable but detectable can be addressed by the algorithm proposed in this chapter, which places mild conditions on each of the physical systems and relaxes the assumptions made in [ME14].

### 6.3.2 Smart Grids

Building on the discussion of the previous section, we introduce the same model for the evolution of the state of a smart power grid as that of [PDB11], namely, a connected power network consisting of  $n$  generators and their corresponding  $n$  generator terminal buses and  $m$  load buses, totaling  $n + m$  buses in the network. The dynamics of the network follow the linear small-signal version of the classical structure-preserving power network model discussed in [SP98], which comprises the dynamic linearized swing equation and the algebraic DC power flow equation. Further details regarding the derivation of such dynamics from the nonlinear model can be found in [Sch04] and [PBB11].

The weighted graph associated with the admittance in the connectivity network induces a Laplacian matrix  $\begin{bmatrix} \mathcal{L}^{gg} & \mathcal{L}^{gl} \\ \mathcal{L}^{gl} & \mathcal{L}^{ll} \end{bmatrix} \in \mathbb{R}^{(n+m) \times (n+m)}$ , where the first  $n$  rows are associated with the buses connecting to the generators and the remaining rows correspond to the bus network.

The whole system can be described by the differential-algebraic continuous-time dynamic model given by

$$N_c \dot{x}(t) = A_c x(t) + u(t) \quad (6.2)$$

where the state  $x = [\delta^\top \omega^\top \theta^\top]^\top \in \mathbb{R}^{2n+m}$ , encompasses the generator rotor angles  $\delta \in \mathbb{R}^n$ , the frequencies  $\omega \in \mathbb{R}^n$ , and the bus voltages angles  $\theta \in \mathbb{R}^m$ . The input term  $u(t)$  accounts for the known changes in input power to the generators or power demands of the loads. The matrices

of the dynamics are as follows

$$N_c = \begin{bmatrix} I & 0 & 0 \\ 0 & N_g & 0 \\ 0 & 0 & 0 \end{bmatrix}, A_c = - \begin{bmatrix} 0 & -I & 0 \\ \mathcal{L}_{gg} & D_g & \mathcal{L}_{gl} \\ \mathcal{L}_{lg} & 0 & \mathcal{L}_{ll} \end{bmatrix},$$

where  $N_g$  and  $D_g$  are the diagonal matrices of the generator inertia and damping coefficients. We assume that the parameters of the network can be estimated as in [CCS11], but, in contrast to [PDB11] where no disturbances and noise are included, we consider the error in the estimation by adding a disturbance term to equation (6.2).

For detection purposes, we assume that a subset of the state variables being measured is corrupted by sensor noise as modeled next. Let  $C \in \mathbb{R}^{p \times n}$  and  $\eta \in \mathbb{R}^p$ , and the signal  $f$  represent cyber-physical attacks in the sensors and/or in the state, leading to the following system equations

$$\begin{aligned} N_c \dot{x}(t) &= A_c x(t) + u(t) + \underbrace{\begin{bmatrix} F & 0 \end{bmatrix} f(t) + E_c d(t)}_{F_c} \\ y(t) &= C_c x(t) + \underbrace{\begin{bmatrix} 0 & L \end{bmatrix} f(t) + v(t)}_{L_c} \end{aligned}$$

where  $F \in \mathbb{R}^{2n+m \times 2n+m}$ ,  $E_c \in \mathbb{R}^{2n+m \times q}$ ,  $L \in \mathbb{R}^{p \times p}$ ,  $d(t) \in \mathbb{R}^q$ ,  $f(t) \in \mathbb{R}^{2n+m+p}$  and both  $F$  and  $L$  are full rank matrices.

The next step is to transform the differential-algebraic system in (6.2) into a standard differential equation model, as described in [PDB11], by resorting to the fact that  $\mathcal{L}_{ll}$  is invertible due to the overall network being connected [Sch04]. This implies that the bus voltage angles  $\theta(t)$  can be obtained from the remaining variables by simply inverting the algebraic equation in (6.2).

If we consider the partition of the matrices  $F = \begin{bmatrix} F_\delta^\top & F_\omega^\top & F_\theta^\top \end{bmatrix}^\top$ ,  $E_c = \begin{bmatrix} E_\delta^\top & E_\omega^\top & E_\theta^\top \end{bmatrix}^\top$  and  $C_c = \begin{bmatrix} C_\delta & C_\omega & C_\theta \end{bmatrix}$ , where the dimensions of the submatrices are in accordance to the state  $x = \begin{bmatrix} \delta^\top & \omega^\top & \theta^\top \end{bmatrix}$ , the following set of equations, known as the kron-reduced system, is obtained

$$\begin{aligned} \begin{bmatrix} \dot{\delta}(t) \\ \dot{\omega}(t) \end{bmatrix} &= \underbrace{\begin{bmatrix} 0 & I \\ -N_g^{-1}(\mathcal{L}_{gg} - \mathcal{L}_{gl}\mathcal{L}_{ll}^{-1}\mathcal{L}_{lg}) & -N_g^{-1}D_g \end{bmatrix}}_{\tilde{A}} \begin{bmatrix} \delta(t) \\ \omega(t) \end{bmatrix} + u(t) + \underbrace{\begin{bmatrix} F_\delta & 0 \\ N_g^{-1}F_\omega - N_g^{-1}\mathcal{L}_{gl}\mathcal{L}_{ll}^{-1}F_\theta & 0 \end{bmatrix}}_{\tilde{F}} f(t) \\ &\quad + \underbrace{\begin{bmatrix} E_\delta \\ N_g^{-1}E_\omega - N_g^{-1}\mathcal{L}_{gl}\mathcal{L}_{ll}^{-1}E_\theta \end{bmatrix}}_{\tilde{E}} \begin{bmatrix} d(t) \\ v(t) \end{bmatrix}, \\ y(t) &= \underbrace{\begin{bmatrix} C_\delta - C_\theta\mathcal{L}_{ll}^{-1}\mathcal{L}_{lg} & C_\omega \end{bmatrix}}_{\tilde{C}} \begin{bmatrix} \delta(t) \\ \omega(t) \end{bmatrix} + \underbrace{\begin{bmatrix} C_\theta\mathcal{L}_{ll}^{-1}F_\theta & L \end{bmatrix}}_{\tilde{L}} f(t) + \underbrace{\begin{bmatrix} C_\theta\mathcal{L}_{ll}^{-1}E_\theta & I \end{bmatrix}}_{\tilde{E}^v} \begin{bmatrix} d(t) \\ v(t) \end{bmatrix}. \end{aligned}$$

Thus, the kron reduced system, with its associated tuple of matrices  $(\tilde{A}, \tilde{B}, \tilde{C}, \tilde{D}, \tilde{E}, \tilde{F}, \tilde{L}, \tilde{E}^v)$ , where  $\tilde{B} = I$  and  $\tilde{D} = 0$ , is in the form of a standard linear time-invariant system, which after the discretization can be written as

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + Ff(k) + Ed(k) \\ y(k) &= Cx(k) + Du(k) + Lf(k) + E^v d(k) \end{aligned} \quad (6.3)$$

where we assume without loss of generality that  $|d_i(k)| \leq 1$  and  $|E^v d(k)| \leq \bar{v}, \forall k \geq 0$ . Notice that system (6.3) has  $n_x = 2n$ . A general discussion regarding the observability of power networks can be found in [TKA<sup>+</sup>12].

For the examples of the power grid in (6.3) and the system of subsystems in (6.1) there is a straightforward solution resorting to the Kalman Decomposition (see Appendix A). In particular, the solution presented in [ME14] is a special case where the transformation for the Kalman Decomposition is constant and depends solely on the structure of the problem. Since the transformation yields a new system with the same transfer function, one can simply apply the Kalman Decomposition, obtain the observable subsystem and design an SVO for the reduced system. Provided that the unobservable modes are stable, convergence is guaranteed and the detection procedure is equivalent to the method provided in this chapter. However, such a solution is troublesome to define for the more general class of LPV systems. For that reason, for a system given by

$$\begin{aligned} x(k+1) &= A_k x(k) + B_k u(k) + F_k f(k) + E_k d(k) \\ y(k) &= C_k x(k) + D_k u(k) + L_k f(k) + E_k^v d(k) \end{aligned} \quad (6.4)$$

we can summarize the fault detection in the following lemma. We point out that in for the coprime factorizations, the system in (6.4) is parameter-dependent but, at each time  $k$ , we have access to the parameter  $\rho$  through measurements and so, no uncertainty is present in the matrices defining (6.4).

**Lemma 6.1** (fault detection). *Consider a dynamic system as in (6.4) and an SVO that produces set-valued estimates,  $X_N(k)$ , for  $x(k)$ , given horizon  $N$  and  $|d(k)| \leq 1, \forall k \geq 0$ . A fault occurred if  $X_N(k) = \emptyset$ .*

There are some major issues using the standard procedure for the aforementioned SVOs: boundedness of the hyper-volume of the sets is only guaranteed if the system is stable with zero input [RSA14] (requiring the system to be observable also yields boundedness of the sets); the computational time associated with the use of the Fourier-Motzkin elimination method which is of intrinsically double exponential complexity; and, for LPV systems where the unobservable components are stable the standard SVOs cannot be used as the estimates diverge. In the previous chapter, we addressed the problem of bounding the necessary horizon value. In the remaining of this chapter, we tackle how to select the coprime factorization as to design SVOs for detectable LPV systems and also how to design SVOs for fault detection without the need to use the Fourier-Motzkin elimination method.

## 6.4 SVOs for detectable systems

An assumption for using SVOs is that the system must be observable or otherwise the produced set grows without bounds. In [RS13], it was proposed the use of the concept of left-coprime factors to bound the horizon required for detection. This result is going to be a building block for faster SVOs (i.e., with diminished computational requirements) in the next section. In this section, we exploit additional characteristics of the coprime factorization to provide a guaranteed rate of convergence of the set-valued state estimates, for the case of detectable systems (i.e. all unobservable modes of the system are stable).

Consider the system (6.4) but where all the exogenous signals are concatenated in  $u$  (and correspondingly for matrices  $B_k$  and  $D_k$ ) so that we get the following dynamics

$$\begin{aligned} x(k+1) &= A_k x(k) + B_k u(k) \\ y(k) &= C_k x(k) + D_k u(k) \end{aligned} \quad (6.5)$$

**Proposition 6.1** (left-coprime factorization [ZDG96]). *Let a discrete-time dynamic system described by (6.5) be detectable, which can be written in a compact matrix notation as*

$$P(k) := \left[ \begin{array}{c|c} A_k & B_k \\ \hline C_k & D_k \end{array} \right]$$

and define

$$\left[ \begin{array}{c|c} S_G(k) & S_Q(k) \end{array} \right] = \left[ \begin{array}{c|c} A_k - K_k C_k & -K_k \quad B_k - K_k D_k \\ \hline R_k C_k & R_k \quad R_k D_k \end{array} \right]$$

where  $R_k$  must always be a nonsingular matrix and  $K_k$  is such that  $A_k - K_k C_k$  is stable. Then,

$$P(k) = S_G^{-1}(k) S_Q(k).$$

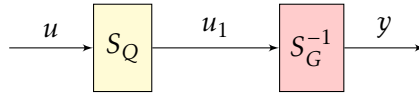


Figure 6.1: Schematic representation of the two coprime systems.

The above factorization is depicted in Figure 6.1. The left-coprime factorization creates two separate systems  $S_Q(k)$  and  $S_G(k)$  and a fault is detected whenever appropriately set-valued estimates for the signal  $u_1$  (see Figure 6.1) for the two systems do not intersect.

The aforementioned technique allows to establish two convergence results for the sets produced by the SVO. If the system is observable, we can select the matrices  $K_k$  such that all eigenvalues of  $(A_k - K_k C_k)^{n_x}$  are equal to zero for any  $k \geq n_x$  with  $n_x$  being the number of states of the system [RSA14], as given in the previous chapter. If the system is detectable, the rate of convergence is governed by the slowest unobservable modes, as shown next.

**Definition 6.1.** *A sequence of sets,  $U(1), U(2), \dots$ , is said to have ultimately bounded hyper-volume if there exist  $\epsilon > 0, k_o \geq 1$  such that  $\text{vol}(U(k)) < \epsilon$  for all  $k \geq k_o$ . Moreover, if  $\text{vol}(U(k)) < \Gamma_o \frac{1-\lambda^k}{1-\lambda}$ , for some  $\Gamma_o, \lambda > 0$ , then the sequence of sets is said to have  $1/\lambda$  convergence.*

The next theorem summarizes the convergence properties of the SVOs. When referring to an SVO producing estimates for the output signal, we mean the set of all points obtained by applying the output equation to any point in the set-valued estimates of its internal state, i.e., for a coprime factor  $S_Q(k)$ , with internal state  $x_Q(k)$ , an SVO will return the set  $X_Q(k)$  such that  $x_Q(k) \in X_Q(k)$  and the estimates for the output  $u_1(k)$ , considering  $R = I$ , is the set defined  $\{u_1(k) : p(k) \in X_Q(k), u_1(k) = C_k p(k) + D_k u(k)\}$ .

**Theorem 6.1** (estimate convergence). *Consider a system  $P$  with dynamic model as in (6.4), with  $f \equiv 0$ , where  $x(k) \in \mathbb{R}^{n_x}$ . Further suppose that a left-coprime factorization as in Proposition 6.1 exists, and that an SVO constructed for  $S_Q(k)$  and  $S_G(k)$ , providing estimates of  $u_1(k)$ , is designed. Finally, assume that  $x(0) \in X(0)$ , and both  $|d_i(k)| \leq 1$ ,  $|v_i(k)| \leq \bar{v}$ . Then:*

- i) *if  $P$  is observable, the hyper-volume of the set-valued estimates of  $u_1(k)$  is ultimately bounded and converge in a finite number of steps;*
- ii) *if  $P$  is detectable, the hyper-volume of the set-valued estimates of  $u_1(k)$  is ultimately bounded with convergence governed by  $\frac{1}{\sigma_{\max}}$ , where  $\sigma_{\max} := \max_{\sigma, k} |\sigma(A_k - K_k C_k)|$ .*

*Proof.* i) The proof can be found in [RSA14] for the LTI case. It revolves around the fact that, for an observable pair  $(A, C)$ , one can place the eigenvalues of  $A - KC$  at the origin and thus obtain a deadbeat observer such that  $(A - KC)^{n_x} = 0$ . For the LPV case, a similar statement is true but for the product of matrices in the last  $n_x$  time instants, i.e.,  $(A_k - K_k C_k)^{n_x} = 0$  (see Chapter 5).

- ii) Since the system is detectable, one can build a state observer satisfying

$$\hat{x}(k+1) = (A_k - K_k C_k) \hat{x}(k) + \begin{bmatrix} L_k & B_k \end{bmatrix} \begin{bmatrix} y(k) \\ u(k) \end{bmatrix},$$

with  $A_k - K_k C_k$  being stable, which means that the state estimate can be written based on the initial estimate as

$$\hat{x}(k) = (A_k - K_k C_k)^k \hat{x}(0) + \sum_{\tau=0}^{k-1} (A_k - K_k C_k)^{k-1-\tau} \begin{bmatrix} L_k & B_k \end{bmatrix} \begin{bmatrix} y(\tau) \\ u(\tau) \end{bmatrix}.$$

Since the system is detectable, take  $\sigma_{\max}$  as defined in the statement of the theorem, which means  $\|(A_k - K_k C_k)^k \hat{x}(0)\| \leq \sigma_{\max}^k \|\hat{x}(0)\|$  and, therefore, an overbound for the set-valued estimate can be written as

$$|\hat{x}(k)| \leq \sum_{\tau=0}^{k-1} \|(A_k - K_k C_k)^{k-1-\tau}\| \left\| \begin{bmatrix} L_k & B_k \end{bmatrix} \begin{bmatrix} y(\tau) \\ u(\tau) \end{bmatrix} \right\| + \sigma_{\max}^k \|\hat{x}(0)\|.$$

Given the exponential rate of convergence associated with the term in  $\hat{x}(0)$  let us look at

the remaining term

$$\begin{aligned}
 & \sum_{\tau=0}^{k-1} \|(A_k - K_k C_k)^{k-1-\tau}\| \left\| \begin{bmatrix} L_k & B_k \end{bmatrix} \right\| \left\| \begin{bmatrix} y(\tau) \\ u(\tau) \end{bmatrix} \right\| \\
 & \leq \sum_{\tau=0}^{k-1} \sigma_{\max}^{k-1-\tau} \left\| \begin{bmatrix} L_k & B_k \end{bmatrix} \right\| \left\| \begin{bmatrix} y(\tau) \\ u(\tau) \end{bmatrix} \right\| \\
 & \leq \frac{(1 - \sigma_{\max}^k)}{1 - \sigma_{\max}} \left\| \begin{bmatrix} L_k & B_k \end{bmatrix} \right\| \max_{0 \leq \tau \leq k} \left\| \begin{bmatrix} y(\tau) \\ u(\tau) \end{bmatrix} \right\|
 \end{aligned}$$

which concludes the proof since the set-valued estimates are bounded and its worst-case is governed by  $1/\sigma_{\max}$ . □

## 6.5 Fast SVOs

In the previous section, a left-coprime factorization was used to obtain a bound on the necessary horizon for the SVO-based fault detection approach, thus eliminating unnecessary computational complexity of considering all past measurements. However, the computational complexity is also tied to the use of the Fourier-Motzkin elimination method to remove the dependence on past instants, which has a doubly exponential complexity. Possible alternatives to the Fourier-Motzkin include any over-approximation technique such as computing hyper-parallelepiped overbounds that introduce conservatism in current estimates.

In this section, new equations to describe the set-valued estimates are provided using the characterization for detectability, which is a reformulation of what is presented in [PDB11], resulting in a *fast* SVO (fSVO) with complexity bounded by the size of the state space,  $n_x$ . In addition, the SVOs mimic the theoretical condition for detectability. In [PBB11] and [PBB12], the technical condition is shown to yield, both in the context of the problem considered here and in consensus problems, zero dynamics in the system for the difference of the input fault signals. This chapter proposes an alternative approach, in the sense that detectability and identifiability of faults are equivalent to the event of the set generated by the SVOs being empty.

Consider the faulty system with no noise, no disturbances, and no other inputs apart from the fault (which to avoid misinterpretations, we label as  $f$  and corresponds to  $u$  in the notation of [PDB11] and [PBB12]). Then, the dynamics in (6.4) become

$$\begin{aligned}
 x(k+1) &= A_k x(k) + B_k f(k) \\
 y(k) &= C_k x(k) + D_k f(k)
 \end{aligned}$$

To include other signals such as the disturbance affecting the state and the noise, we can use  $\begin{bmatrix} f(k) \\ d(k) \end{bmatrix}$  and replace accordingly the matrix  $F_k$  by  $\begin{bmatrix} F_k & E_k \end{bmatrix}$  and  $L_k$  by  $\begin{bmatrix} L_k & E_k^v \end{bmatrix}$ .

By resorting to the factorization in Proposition 6.1 and the results in Theorem 6.1, i.e., convergence in finite-time if the system is observable or an asymptotic rate of convergence if the

system is detectable, it is possible to remove the use of the projection step. The main advantage is avoiding the Fourier-Motzkin elimination method at the expenses of not maintaining an estimate for the current state.

In the construction of the proposed SVO, it is helpful to introduce the definitions of fault detectability and fault identifiability.

**Definition 6.2** (fault detectability [PDB11]). *Consider a system with model given by (6.5) and a fault profile  $f_1(k), 0 \leq k \leq k_t$ . A fault  $f_1$  is detectable in  $k_t$  time instants if there does not exist  $x(0) \in \mathbb{R}^{n_x}$  that satisfies*

$$C_k A_k^k x(0) + \sum_{\tau=0}^{k-1} C_k A_k^{k-1-\tau} F_k f_1(\tau) + L_k f_1(k) = 0 \quad (6.6)$$

for all  $0 \leq k \leq k_t$ .

Notice that (6.6) can be rewritten in vectorial form as

$$\begin{bmatrix} C_k \\ C_k A_k \\ \vdots \\ C_k A_k^{k_t} \end{bmatrix} x(0) = \begin{bmatrix} -L_k f_1(0) \\ -C_k F_k f_1(0) - L_k f_1(1) \\ \vdots \\ -\sum_{\tau=0}^{k_t-1} C_k A_k^{k_t-1-\tau} F_k f_1(\tau) - L_k f_1(k_t) \end{bmatrix}.$$

We also introduce a similar definition regarding the identifiability of the faults.

**Definition 6.3** (fault distinguishability). *Take a system with model given by equation (6.5) and a fault profile  $f_2(k), 0 \leq k \leq k_t$ . Fault  $f_2$  is distinguishable in  $k_t$  time instants from fault  $f_1$  if there does not exist  $x(0) \in \mathbb{R}^{n_x}$  that satisfies*

$$\begin{bmatrix} C_k & L_k^{f_1} & 0 & \cdots & 0 \\ C_k A_k & C_k F_k^{f_1} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ C_k A_k^{k_t} & C_k A_k^{k_t-1} F_k^{f_1} & \cdots & C_k F_k^{f_1} & L_k^{f_1} \end{bmatrix} \begin{bmatrix} x(0) \\ f_1(1) \\ \vdots \\ f_1(k_t) \end{bmatrix} = \begin{bmatrix} -L_k f_2(0) \\ -C_k F_k f_2(0) - L_k f_2(1) \\ \vdots \\ -\sum_{\tau=0}^{k_t-1} C_k A_k^{k_t-1-\tau} F_k f_2(\tau) - L_k f_2(k_t) \end{bmatrix}.$$

In the above definitions, for two different fault signals  $f_1$  and  $f_2$ , we define  $x(0) = x_1(0) - x_2(0)$  where  $x_1(0)$  and  $x_2(0)$  are the initial conditions for the system using  $f_1$  and  $f_2$  respectively.

In order to perform fault detection, we will have to consider the nominal “fault-free” model and distinguish it from the actual system for which we have measurements  $y(k)$ . Considering model (6.4) with disturbances and noise signals and using the above definitions, we can rewrite the SVO equations so as to make all the inequalities be written using a single time instant, i.e., all inequalities pose constraints on the  $x(k - N)$  variable. The new set of inequalities for the SVOs are:

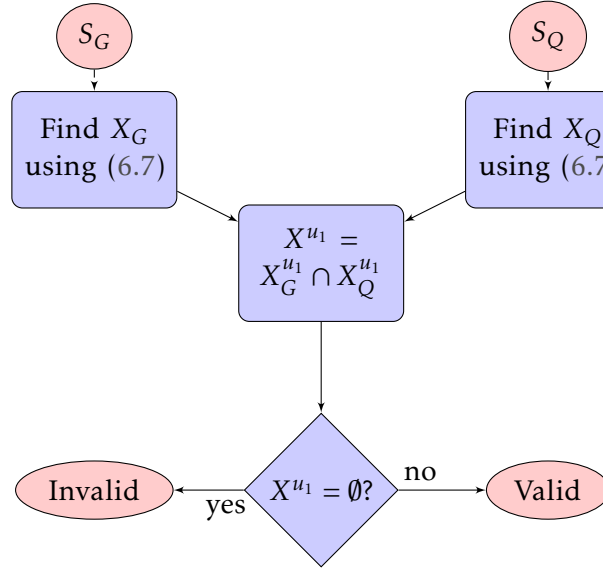


Figure 6.2: Flowchart of an iteration of the Fast SVO algorithm which takes as input the coprime factorization and decides if the model is invalid or still valid.

$$\begin{bmatrix} \bar{C}_k & \bar{L}_k & 0 & \cdots & 0 \\ \bar{C}_k A_k & \bar{C}_k F_k & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \bar{C}_k A_k^n & \bar{C}_k A_k^{n-1} F_k & \cdots & \bar{C}_k F_k & \bar{L}_k \\ 0 & \bar{I} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \bar{I} \\ M(k-N) & 0 & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} x(k-N) \\ d(k-N) \\ \vdots \\ d(k) \end{bmatrix} \leq \begin{bmatrix} \bar{y}(k-N) \\ \vdots \\ \bar{y}(k) \\ 1 \\ \vdots \\ 1 \\ m(k-N) \end{bmatrix} \quad (6.7)$$

The above SVO equation is no longer an iterative solution since it is not possible to obtain estimates for  $x(k-N)$  and obtain  $M(k-N)$  and  $m(k-N)$ . Nevertheless, we can assume a sufficiently *large* set and use the coprime factorization presented in the previous section to remove the conservatism of that overbound. In essence, (6.7) will be applied to the LPV models of the coprime factors of (6.4) by replacing the matrices according to Proposition 6.1.

The Lemma 6.1 can be reformulated for the novel SVO equations obtaining:

**Lemma 6.2** (fault detection). *Consider a dynamic system as in (6.4) and an fSVO defining the inequalities for  $x_Q(k-N)$  and  $x_G(k-N)$  of the coprime factors  $S_Q(k)$  and  $S_G(k)$  given by Proposition 6.1 with sufficiently large sets such that  $x_Q(k-N) \in X_Q(k-N)$  and  $x_G(k-N) \in X_G(k-N)$ . A fault occurred if there is no solution to the inequalities (6.7) for both coprime factors.*

Lemma (6.2) comes directly from the fact that if there are no solutions to (6.7) the “fault-free” model does not correspond to the real system. Fault isolation can be performed by invalidating the models for all the remaining faults. For each of the  $\ell$  considered faults, we define pairs of matrices  $(F_k, L_k)$  such that only that fault is modeled, thus creating  $\ell$  possible models for the system. If the faults are identifiable, then all SVOs become empty except for the one which



represents the correct fault model. If multiple faults are to be considered we could use a scheme such as the one presented in [BRSO15].

A decision regarding the model for each possible fault being compliant with the measurements is made based on the algorithm presented in Figure 6.2. The sets  $X_G^{u_1}$  and  $X_Q^{u_1}$  denote, respectively, the set of possible values of output  $u_1$  for the cofactor system  $S_G(k)$  and  $S_Q(k)$ . Testing if  $X^{u_1}$  is the empty set amounts to solving a feasibility program of existing a point in  $X_G$  and another in  $X_Q$  such that the outputs of the subsystems  $S_G(k)$  and  $S_Q(k)$  are the same.

An important issue regarding the fSVOs is that they are not suitable for state estimation. As all the inequalities are written with respect to  $x(k - N)$ , no estimates are available for  $x(k)$ . In the standard SVOs, restrictions concerning the state in all last  $N$  iterations are then projected to depend solely on the current time instant. Following this reasoning, no iterative computation of the set-valued estimates is possible, which makes them not suitable for state estimation. In addition, following the factorization, the two SVOs for subsystems  $S_Q(k)$  and  $S_G(k)$  have states that are internal to each of the subsystems and, therefore, are not related to the original system state  $x(k)$ . Nevertheless, for applications such as fault detection and isolation and model invalidation, they are suitable as the state itself may be disregarded.

## 6.6 Simulation Results

In this section, we present a set of simulations illustrating the fault detection mechanism described in this chapter. In particular, we are interested in comparing against the approach of performing a canonical Kalman decomposition, which is valid only for the LTI case whereas our proposal addresses the broader class of LPV systems. This distributed fault detection architecture reduces the dependability on a single centralized point of detection, offering a more robust fault detection strategy but increasing the aggregated computational power, since each node acts itself as a detector. We start by analyzing the example described in [ME14] which resorts to the Kalman decomposition for a particular example.

Recovering the example, each subsystem is a flexible link robot dynamic system modeled as:

$$\begin{bmatrix} \dot{\theta}_m^i \\ \dot{\omega}_m^i \\ \dot{\theta}_\ell^i \\ \dot{\omega}_\ell^i \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{K_\ell}{J_m} & -\frac{B}{J_m} & \frac{K_\ell}{J_m} & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{K_\ell}{J_\ell} & 0 & -\frac{L_\ell}{J_m} - \frac{mgh}{J_\ell} & 0 \end{bmatrix} \begin{bmatrix} \theta_m^i \\ \omega_m^i \\ \theta_\ell^i \\ \omega_\ell^i \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{K_\tau}{J_m} \\ 0 \\ 0 \end{bmatrix} u^i + \begin{bmatrix} 0 \\ \frac{K_\tau}{J_m} \\ 0 \\ 0 \end{bmatrix} f^i + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{mgh}{J_\ell} \end{bmatrix} d^i$$

$$y_i = \sum_{j \in \mathcal{N}_i} C(x_i - x_j)$$

for  $i \leq \mathcal{S}$  and  $C = [I_3 \ 0_{3 \times 1}]$ . The states represent the angular position and velocity of the motor shaft ( $\theta_m^i$  and  $\omega_m^i$ ), and the angular position and velocity of the link ( $\theta_\ell^i$  and  $\omega_\ell^i$ ). For further details on the subsystems dynamical models, the interested reader is referred to [ME14] and the references therein. The network topology is selected at random in each time instant with 25 nodes, a minimum and maximum degree of the interconnection graph of 1 and 3, respectively.

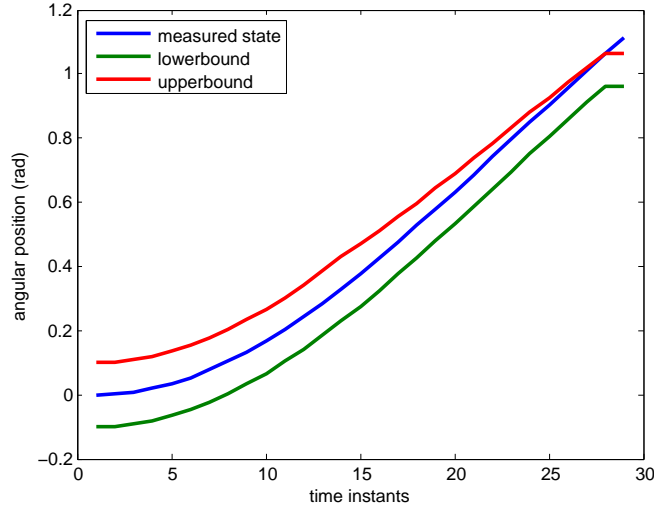


Figure 6.3: Example of a simple fault detection where the state of the system (blue line) crosses the upperbound (red line) of the state given through the projection of the set-valued estimate onto the corresponding coordinate.

We assume that the topology is available to the nodes so that the parameter  $\rho$  can be determined. The system is discretized using a sampling time of 0.01 seconds, and the simulations are run for 100 discrete time steps. The simulations displayed are the result from the computations at node 1.

We consider three different scenarios: one where a subsystem has an actuator fault represented by a constant fault signal; a second where this fault is random across time; and a last one where no fault is injected, but the predefined bounds for the disturbance are not satisfied. Each scenario aims to illustrate a different aspect of the detection algorithm.

We start by considering a fixed topology so that the system becomes a Linear Time-Invariant (LTI) model. The Kalman decomposition is performed by applying the state transformation specific to the case of subsystems with relative measurements [ME14]. A standard SVO is then designed for the observable subspace. The aim is to show that detection for this case is possible although the conservatism is not removed since we did not perform the coprime factorization. Figure 6.3 depicts the detection of the algorithm using SVOs. The red and green lines represent the upper and lower bounds of the state variable for the angular position of node 1. These are obtained by projecting the set of estimates onto the coordinate corresponding to this variable. When the state of the system crosses one of the bounds, the corresponding observation will produce an empty set, as none of the admissible state realizations is compatible with the input/output sequences.

In the simulation, we also designed the SVOs for the coprime factors obtained from the system corresponding to the observable subspace and compared it with the proposed method of designing the SVOs for the original detectable system. The two strategies produced the same results for the LTI case. However, the Kalman decomposition is defined only for LTI systems and, therefore, one of the advantages of the proposed technique is to make it possible to construct

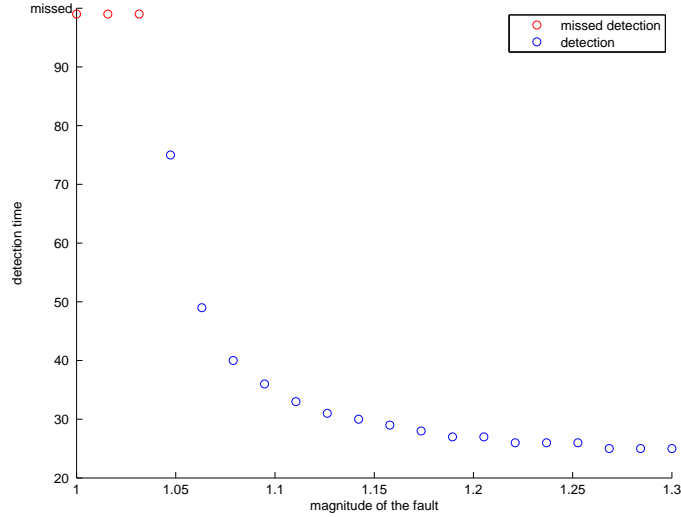


Figure 6.4: Reported detection times when varying the magnitude of a constant fault.

the SVOs *oblivious* to unobservable modes as long as they are stable.

The next simulations were conducted using the proposed factorization-based implementations of the SVOs for the LPV system. In Figure 6.4, it is shown the detection time for the case of a constant actuator fault, as a function of the associated amplitude. As soon as the magnitude of the signal rises slightly above the bound considered for the disturbances, the fault is detected, as the model with  $f = 0$  is not able to generate state realizations compatible with this fault.

Based on the previous results, we investigate fault profiles that hinder detection. Intuitively, the faults harder to detect are likely to behave as modeled disturbances. Following this reasoning, we consider the case where the fault is stochastic with uniform distribution with support on the interval from zero to the maximum magnitude in order to determine its impact. Figure 6.5 shows the mean detection time for the simulated case for a Monte Carlo experiment with 1000 runs. It is noticeable that the detection requires a higher magnitude than the constant case, due to the fact that the fault signal magnitude is lower than the deterministic case most of the times.

It is stressed that the SVOs provide means to tackle a wide range of models for the dynamic system. We take advantage of this fact to further evaluate the proposed method in a more demanding scenario. By definition, every fault is going to be detected as long as the measurements do not comply with the assumed fault-free model. For this reason, we introduced unmodeled stochastic uniformly distributed disturbances, with support on the interval from zero to the maximum magnitude, to the state of the system. Notice that in the dynamics of the subsystems, the disturbances only affect the variable  $\omega^{i_\ell}$  which makes the detection troublesome.

Figure 6.6 shows the mean detection times for the unmodeled disturbances case for a Monte Carlo experiment with 1000 runs. The fault is only detected when its maximum magnitude reaches above 1, which is clearly above the previous required magnitude values for the fault

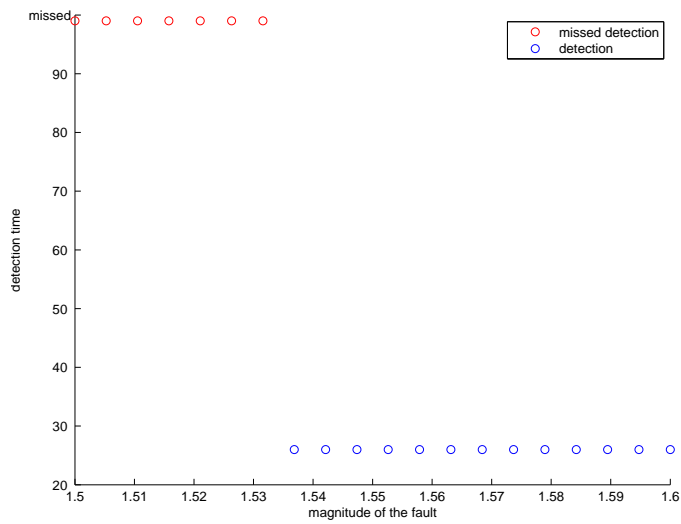


Figure 6.5: Mean detection times when varying the maximum magnitude of a random fault.

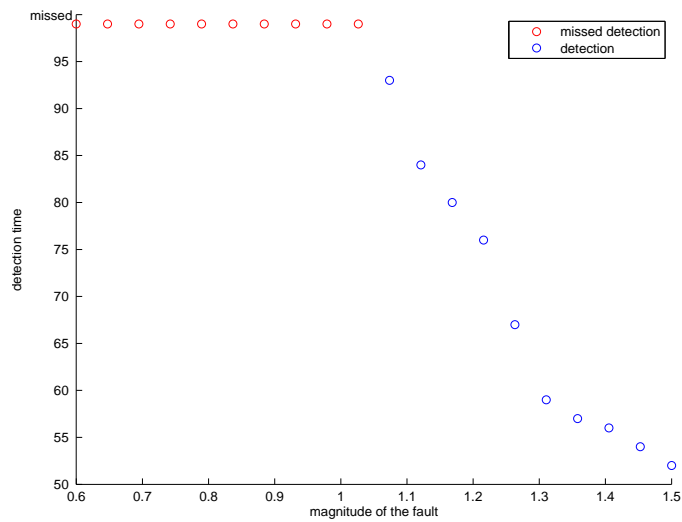


Figure 6.6: Reported detection time for a fault free system but with unmodeled disturbances.

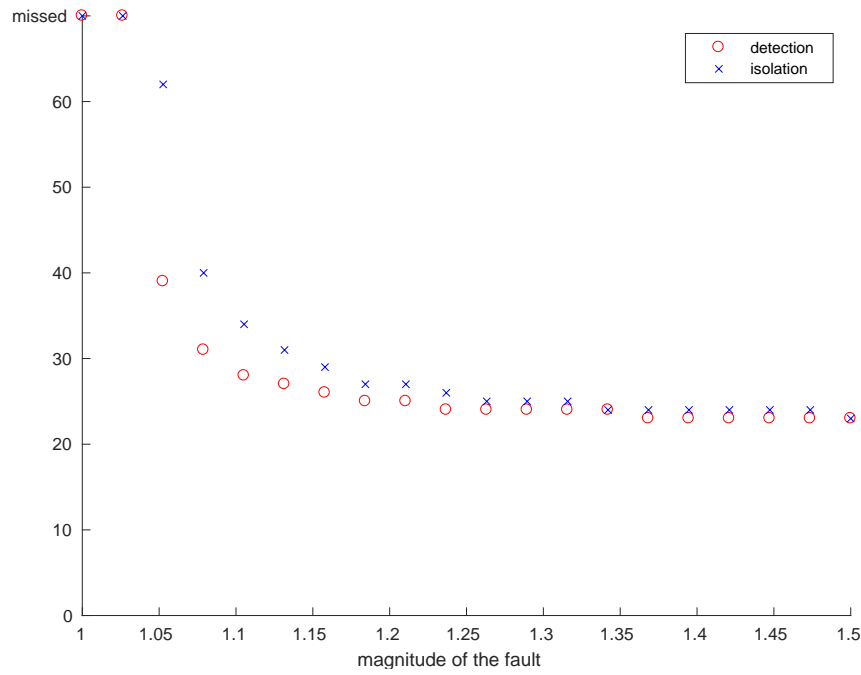


Figure 6.7: Detection and isolation of fault  $f_1$  in the system.

signal.

The fault isolation scheme was also simulated using the aforementioned example. Two different faults were considered, namely  $f_1(k) := [c \ 0]^T$  and  $f_2(k) := [0 \ c]^T$ , for a varying constant  $c$ . The simulation run 3 SVOs: an SVO for the “fault-free” model for fault detection; another that considered  $B_k f_1(k)$  for determining that  $f_1$  is not the current fault; and, a similar to the latter but considering  $f_2(k)$ . Fault detection means the first SVO produced an empty set and upon that event, isolation of the faults is determined when only one of the SVOs is not producing the empty set. In this simulation, after 20 time instants, fault  $f_1$  is injected in the system.

Figure 6.7 reports the detection and isolation times for fault  $f_1$ . We point out that the constant  $c$  cannot be directly compared with the bound for the disturbances without taking into account the small values in matrices  $B_k$ . Figure 6.7 presents 1000 montecarlo runs, but is interesting that in some of the runs, isolation (i.e., SVO for  $f_2$  reports the empty set) happens before the detection as both fault signals have different directions and contribute to a quicker violation of the bounds for the disturbances.

A last important feature of the algorithm is its convergence, which we showed to depend on the slowest unobservable mode if the system is detectable. In Figure 6.8, we present the bounds given by the SVO when the disturbance and noise signals were equal to zero and without performing the coprime factorization. The tight bounds mean the SVOs produce sets that converge when the system is observable and with no disturbance and noise signals. However,

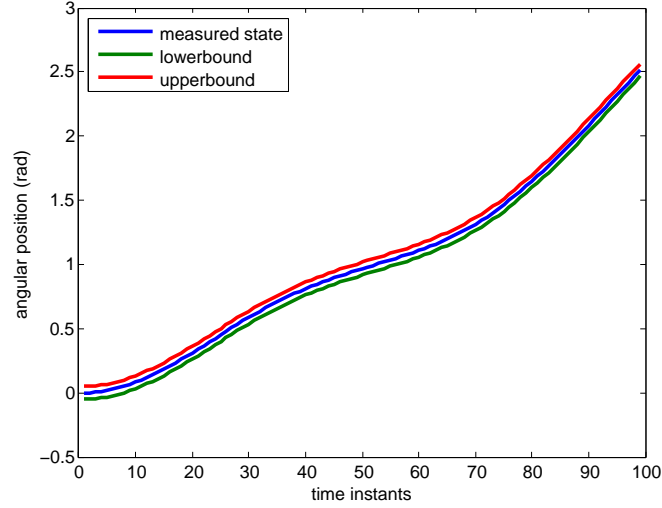


Figure 6.8: Lower and upper bounds of the set-valued estimates when not in the presence of disturbances.

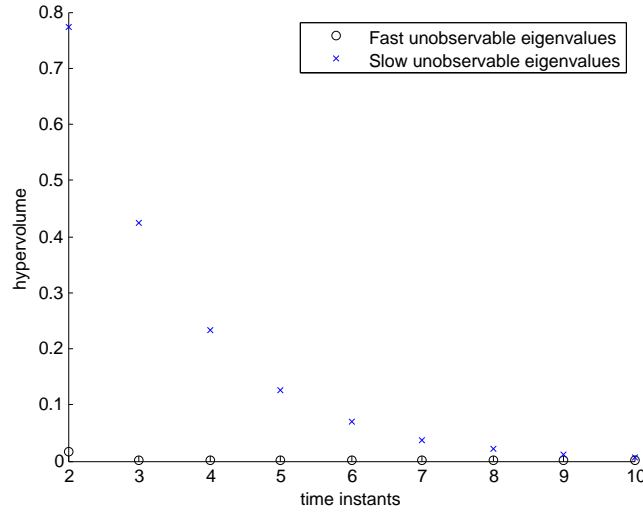


Figure 6.9: Hypervolume of the set corresponding to the system  $S_G$  for eigenvalues of  $A - KC$  close to zero (deadbeat) and with  $\lambda_{\max} = 0.74$ .

the coprime factorization yields the same characteristics when the model is corrupted by noise and unknown but bounded disturbances.

In order to illustrate our main result for detectable systems, we simulated a simple example where by construction the system is unobservable but we can tune the eigenvalues of the dynamics matrix. Consider the dynamic system given by

$$A = \begin{bmatrix} \lambda_{\max} & 1 \\ 0 & -\frac{1}{2} \end{bmatrix}, B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, C = \begin{bmatrix} 0 & 4 \end{bmatrix}, D = 0,$$

where by selecting the value of  $\lambda_{\max}$ , we tune the unobservable mode. For this example of a system with two states, it is always unobservable and we resort to the left-coprime factorization proposed in [RPK92]. In Figure 6.9, it is depicted the hypervolume of the sets for the case of fast unobservable modes (selecting  $\lambda_{\max} = 0$  yields eigenvalues of  $A - KC$  equal to zero and slow

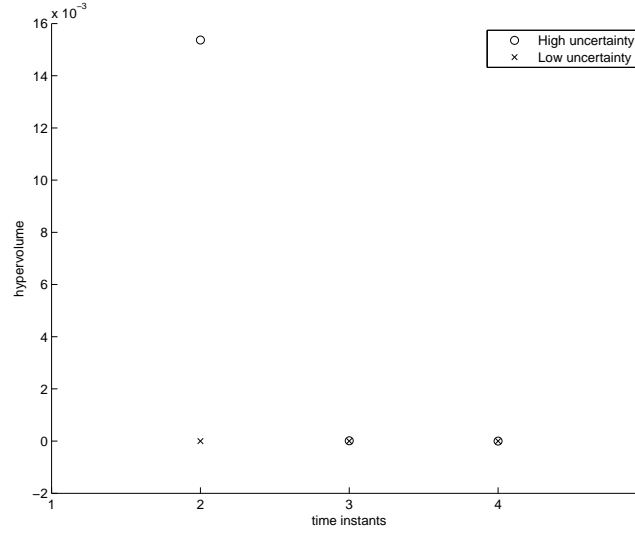


Figure 6.10: Hypervolume of the set corresponding to the system  $S_G$  for eigenvalues of  $A - KC$  close to zero (deadbeat) and uncertainty of 1 and  $10^6$  for the initial state.

unobservable eigenvalues of the term  $A - KC$ , by setting  $\lambda_{\max} = 0.74$ ). For the fast unobservable modes case, the size becomes constant after 2 time instants corresponding to the size of the state space and following the results in Theorem 4.2. When the eigenvalues are slow, the convergence is asymptotic, in the sense of Definition 3.6, and slower when compared to the fast unobservable modes case.

The key point to note when using the fSVOs is that the uncertainty of the initial state is removed after the horizon as given by Theorem 4.2. This feature is crucial for the procedure since by not having an iterative algorithm there is no available estimate for the state  $x(k - N)$ . However, Figure 6.10 illustrates the results for the fast unobservable modes case when we start with an uncertainty equal to a hypercube of side 1 and  $10^6$ . As expected, after two time instants the size of both set-valued estimates for the internal state of system  $S_G$  are equal and remain constant for the rest of the simulation. In Figure 6.11, we present the median and quartiles (25% and 75%) for the running time of a single iteration of the SVOs against the fSVOs for 1000 runs. It is worth pointing out that the small horizon and the fact that we are using hyper-parallellepiped approximations for the projections already make the SVOs considerably efficient. Nevertheless, fSVOs still reduce the computational time to almost half when compared to the SVOs using the same coprime factorization.

We also simulated the smart power grid network case, as another example of a cyber physical system. We consider the well-known test bed example IEEE 14 bus system [oW15], which is depicted in Figure 6.12. We assume a sampling period of 1 second and run the simulations for 20 seconds.

The first simulation results illustrate the equivalency between the theoretical condition for fault detection and the use of an SVO without any disturbances. Figure 6.13 depicts the rotor

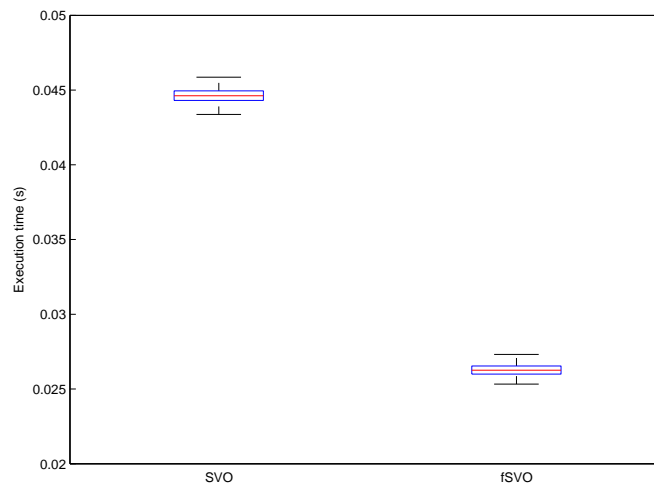


Figure 6.11: Running time of the SVOs compared with the fSVOs.

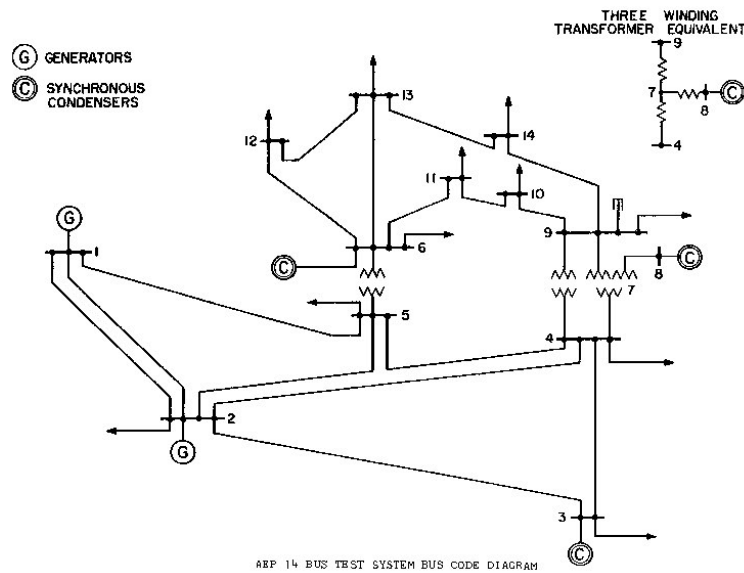


Figure 6.12: IEEE 14 bus system test bed example [oW15]



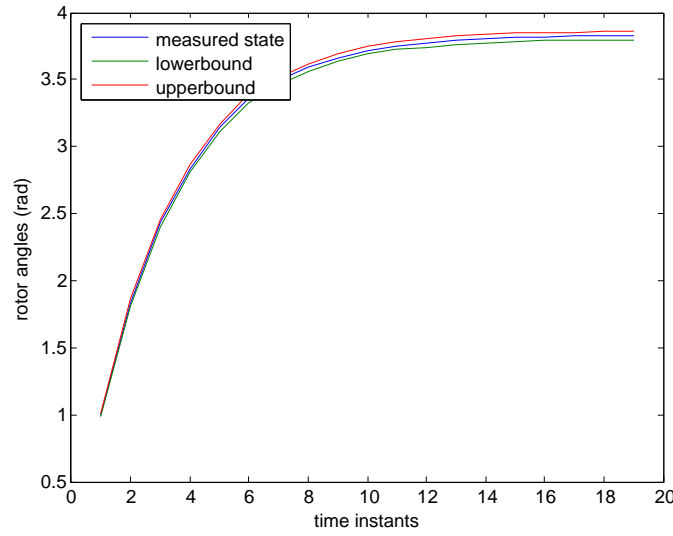


Figure 6.13: SVO tracking of the true state of node 1 in the network.

angle of node 1, which presents a similar behaviour to the remaining generator rotor angles across the network. In the computation, and in order to avoid numerical errors, we introduce an oversizing [Ros11] of the set by a factor of  $10^{-2}$  and that is what leads to the difference between the lower and upper bounds presented in Figure 6.13. As we described in the previous section, the SVOs replicate the behavior of the plant and the detection is exact if there are no disturbances or uncertainties. The horizon was set to  $N = 20$ , in order to be larger than the number of states of the network.

Another interesting point to be illustrated is how SVOs can isolate faults. In the former simulations the ability of the method was presented for fault detection and a simple strategy would be to design an SVO for each of the faults and when only one SVO is active, the fault is identified. However, such procedure entails a combinatorial number of SVOs. As an example, for 5 generators and assuming a maximum of 2 simultaneous faulty generators would require 15 SVOs (one for each single generator failing and one for each pair of faulty generators). We now illustrate that by designing an SVO aggregating generator faults, it is possible to identify faults resorting to fewer SVOs. We consider two SVOs: 1) all generators are injecting random signals; 2) generators 1, 2 and 3 do not suffer any fault and their rotor angles are not corrupted.

In simulation, a fault was reported after two seconds by SVO 2 (i.e., its estimated set was empty), which means a fault occurred in at least one of the first 3 generators. By applying iteratively this method it is possible to isolate a fault by constructing SVOs using the past measurements and perform a binary search over the possible faults. If we assume only 1 generator can fail at a time, we need  $\lceil \log_2 n \rceil$  steps in the binary search and design two SVOs at each step mapping half of the faulty nodes. For the general case of  $F$  possible faulty generators, we have the expression  $\lceil F \log_2 n \rceil$ .

### 6.7 Conclusions

This chapter addressed the problem of detecting faults in a distributed environment when the overall system of systems has stable unobservable modes (i.e. it is detectable). Traditional SVOs require observability or the estimates can diverge meaning that the hypervolume of the produced set-valued estimates can tend to infinity as time progresses. In addition, SVOs include operations that are very costly in terms of computational time, which diminishes their applicability to time-sensitive plants.

Nevertheless, SVOs were adopted due to their ability to cope with asynchronous measurements and allow general models that can incorporate both physical systems and their interconnection with networks. By performing a left-coprime factorization, we were able to show that these observers can also be designed for detectable systems with guaranteed convergence rates for the estimates. These are of prime interest as they mean that conservatism in prior estimates has an effect that is at least exponentially going to zero. Building on this result, we are able to rewrite the equations of the SVOs to mimic the theoretical conditions for fault detectability and identifiability and, therefore, avoid the use of the Fourier-Motzkin elimination method, as the whole set was written in terms of a fixed time instant, thus speeding up the computations. The initial uncertainty also vanished due to the convergence property of the estimates.

Simulation results have shown that when the maximum magnitude of a fault exceeds the disturbance bounds, the detection occurs and the time before declaring the faulty state goes to near the size of the state space. Both constant and stochastic faults were simulated using a group of flexible link robots models. In addition, the SVOs were capable of detecting unmodeled disturbances and declare faults whenever the model was not compliant with the measurements. Resorting to another application to smartgrids, we verify the effectiveness of the detection procedure for cyber-physical systems.

## EVENT- AND SELF-TRIGGERED NCS AND SET-VALUED OBSERVERS

### 7.1 Introduction

In the context of distributed systems and Networked Control Systems (NCSs), the performance bottleneck is often located in the communication network, either due to low bandwidth, competition for access to a shared medium of communication, or because the network is much slower than the remaining components of the control loop. In distributed systems, different nodes are typically running an algorithm to achieve a certain goal and are often designed to use information from their communicating neighbors. In networked control systems, sensors might be spatially spread over a region of interest and, therefore, measurements have to be sent to a controller/observer over the network. In any of such cases, the network resources are valuable and the communication issues must be considered as they can prevent the stability as given in [WYB02] and [ZMXZ15]. For further details on this topic, the reader is referred to the detailed survey in [TC03], [HNX07], [ZHY16] and [GYH17]; and the book [BHJ10].

In the control community, two main strategies have emerged to reduce the communication overhead, namely: event triggering, where the sensor decides, based on the current measurements, if it should transmit to the controller/observer the measured quantities; self triggering, where the controller/observer decides, based on the current estimate of the state, when the sensor should perform the next measurement. An event-triggered solution results in a more informed choice, since the sensor has access to the actual measurement, but prevents the sensor from being shut down between updates. For a recent discussion on event- and self-triggered control and estimation, the reader is referred to [HJT12].

The problem of state estimation for general discrete-time Linear Parameter-Varying (LPV) systems relates to that of determining the set of possible future state values for a given set of inputs, initial state, measurements, and (deterministic) bounds on the noise and disturbances affecting the system. LPV models allow for considering NCSs with parametric uncertainty that may arise from incomplete knowledge of the physical parameters of the processes to be

controlled. In the context of distributed observer-based control strategies, uncertainty may also arise due to node heterogeneity or the inability to determine at the observer side which nodes are communicating or taking actions upon the plant. Two interesting instances of the state estimation problem can be found in the following contexts:

**Asynchronous distributed algorithms** determining the state of each of the nodes given partial measurements and knowledge of the whole system dynamics;

**Networked control systems** the observer must generate an estimate of the state and decide when to require a sensor update or define event conditions for the sensors to take that decision.

Throughout this chapter, we focus on two main applications of the theoretical developments of the SVO, namely the application of SVOs to obtain set-valued state estimates of event- and self-triggered networked control systems; and their use for fault detection in randomized distributed systems. They are intrinsically related in the sense that, in both cases, the goal is to minimize either the sensor updates or the computational burden associated with the set-valued computations, in order to reduce the overall cost of implementation of this method in such systems.

Fault Detection and Isolation (FDI) has been a long-standing research topic, since the early 70's (see [Wil76]), but still poses remarkable challenges to both the scientific community and the industry (see, for example, the survey in [HKKS10] and references therein). Classical fault detection methods such as the ones proposed in [Wil76], [BB04], [Duc09] and [NVR08], rely on the design of filters that should be able to generate *large* enough residuals under faulty environments. These strategies aim to derive bounds (or thresholds) on these residuals that can be used to decide whether a fault has occurred or not. However, the calculation of these thresholds is typically cumbersome or poses stringent assumptions on the exogenous disturbances and measurement noise acting upon the plant. In contrast, SVOs aim to compute a set-valued estimate of the state under mild assumptions such as the existence of an overbound for all the signals in the system.

In the context of fault detection, focus is given to the special case of randomized distributed algorithms, for two reasons: their relevance in certain problems — applications range from selection and sorting [MR10] to consensus [BGPS06] and solving high-complexity problems; and, because of their unstructured nature, i.e., all nodes play the same role in the algorithm, while the messages need not satisfy any particular type of time sequence, since any two messages are regarded as having the same purpose. Detecting faults in a distributed way in this setup may lead to a persistent computational and communication overhead, while a self- or event-triggered strategy may yield similar results with far fewer computational and network requirements by running the procedure to obtain the set-valued estimates only when the updates can contribute to the detection.

Within the aforementioned framework, this chapter is concerned with obtaining set-valued estimates of the state of the system that are guaranteed to contain the actual state. The approach of using Set-Valued Observers (SVOs) first introduced in [Wit68] and [Sch68], is adopted - further details can be found in [Sch73], [MV91] and the references therein. The SVO paradigm has the advantage of posing mild assumptions on the system, while allowing for the computation of *a priori* bounds for the maximum error. However, the computational cost is still one of the main issues associated with using SVOs (see [CRS15]). In the remainder of this chapter, this limitation will also be tackled by resorting to the use of event- and self-triggered strategies.

The adoption of a mathematical formulation for representing the set of possible states entails the need for fast and non-conservative intersections and unions of sets, as those are the major time-consuming operations when implemented in a computer. An alternative would be to use the concept of zonotopes, described in [BR71] and further developed in [Com05] and [ABC05]. However, these represent a different compromise between the speed of the unions and intersections, with the intersections requiring more computations and introducing conservatism. Alternatively, the idea of interval analysis [Moo66] may also be adopted, although it introduces conservatism by not considering higher horizon values in their formulation, unlike the SVOs [RS13]. Any set-based approach differs from other methods, such as those employing, for example,  $H_\infty$  filters [WQKW14], in that it provides all possible values of the system state that are compatible with the measurements, which is ideal for the implementation of event-triggered strategies obviating the need for defining threshold values, because the set itself produced by the SVOs represents the event condition for triggering an update.

The strategy for an observer to self-trigger a sensor measurement based on its estimates can resort to an optimization over the update patterns such as in [AH14], where the disturbances and noise are assumed to be Gaussian. In [ASP14] and [ASP15], Kalman-like filters are proposed for state estimation, thus not providing a deterministic bound for the error. For event-triggered systems, a triggering condition can be posed on the norm of the estimation error being below a given threshold, dependent on the norm of the state [MT11] [ASP17]; requiring the derivative of a Lyapunov function of the state being semi-negative definite [HJT12], [MT08]; or, having the norm of the state below a certain threshold [HSB08].

The aforementioned methods can be organized into three groups: algorithms that decide on when to transmit data based on some information about the probability distribution of the state (i.e., using, for example, the covariance matrix produced by the Kalman filter); methods that run an optimization over the possible trigger patterns; and algorithms that perform that decision based on some energy measure of the state (Lyapunov-like and norms under thresholds). These three categories of solutions differ from the current proposal. For the first type, the main difference lies on the use of worst-case set-valued estimates instead of probabilistic filters. The second one involves a complex optimization, as opposed to the *greedy* approach proposed for the SVOs. The last group of solutions based on norms or Lyapunov functions often differ from a

polytopic definition for triggers in the sense that compromise accuracy to gain in performance.

From the perspective of computational load, a Kalman filter solution is attractive due to its light complexity, but does not provide worst-case guarantees as the decision relies on the probability distribution of the state. In addition, designing event-triggered strategies is a non-trivial task, since triggering is based on a threshold imposed to the variance and not on the particular measurements. The remaining strategies revolve around the concept of measuring the energy of the state in some way. These are connected to an SVO-based approach in the sense that both define sets of admissible state values and otherwise a trigger is generated. There is an inherent trade-off between accuracy and complexity. In particular, for LPV systems, a better accuracy provided by the SVOs represents a higher computational cost, but it might also enable a triggering strategy that demands fewer sensor updates.

In this chapter, event- and self-trigger strategies are investigated for networked control systems with the objective of developing an online strategy based on set-valued estimates, which means that, at each time instant, the observer produces a polytope, to which the state is guaranteed to belong, and either triggers or allows the sensor to decide the next time instant to perform a measurement update.

The class of problems herein addressed poses challenges to the state estimation scheme since, due to the random behavior of gossip algorithms or the network medium, for each possible sensor transmission, the state can belong to a set of possible state realizations originated by the dynamics and the previous state. To consider the worst-case scenario, one needs to perform the union of all possible state sets, which, in general, returns a non-convex set [RSA14]. Furthermore, the number of sets grows exponentially with the number of past time instants considered, i.e., the horizon  $N$ . As a result, appropriate tools must be employed to reduce this complexity.

## 7.2 Main Contributions and Organization

The main contributions of this chapter, presented in the papers [SRHS15c] and [SRHS18], can be summarized as follows:

- Given a specific structure for the matrix defining the polytope (i.e., the set-valued state estimate), it is shown how to compute an overbounding hyper-parallelipiped, ellipsoid, or ball;
- Based on the concept of singular vectors, we show how a rotation can be found to prevent the approximation error of using boxes from going to infinity when the matrix defining the polytope is ill-conditioned;
- For the special case of a distributed linear algorithm with a gossip property, it is shown that the overbounds are efficient to compute and propagate, since its complexity is constant;

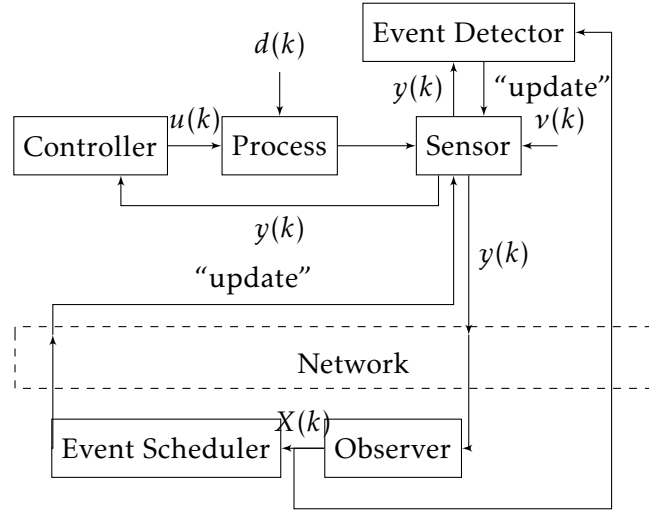


Figure 7.1: Block diagram of a NCS. The Event Detector and Event Scheduler blocks implement event- and self-triggered strategies, respectively, based on the set denoted by  $X(k)$  produced by the observer.

- It is described how the set-valued state estimates provided by the SVOs can be used to define event- and self-triggering conditions for NCS;
- An algorithm is introduced that uses overbounding methods to approximate the *optimal* SVO estimates, which is less computationally demanding, and event- and self-triggers the computation of the aforementioned estimates only when necessary to ensure convergence;
- Results are provided regarding the worst-case frequency of the triggers for a class of LPV systems and its probabilistic counterpart when the distribution of the model uncertainties is known *a priori*;
- Finally, it is given an improved result for convergence that takes into consideration the structure of the output equation of the LPV system.

### 7.3 Problem Statement

In this section, we address the problem of estimating the set of possible state values for a distributed system or a networked control system (see Figure 7.1 for an illustration), which can be described by a discrete-time Linear Parameter-Varying (LPV) system of the form:

$$\begin{cases} x(k+1) = A(\rho(k))x(k) + B(k)u(k) + L(k)d(k) \\ y(k) = C(k)x(k) + v(k) \end{cases} \quad (7.1)$$

with bounded unknown exogenous disturbances,  $d(k) \in \mathbb{R}^{n_d}$ , bounded unknown sensor noise,  $v(\cdot)$ , and uncertain initial state  $x(0) \in X(0)$ , where  $X(0)$  is a known polytope. It is assumed that  $x(k) \in \mathbb{R}^{n_x}$  and the known exogenous input vector  $u(k) \in \mathbb{R}^{n_u}$ . Without loss of generality  $|d_i(k)| \leq 1, \forall k \geq 0$  and  $|v_i(k)| \leq v^*$ . The dynamics matrix,  $A(\cdot)$ , is affine on the polytopic on the unknown parameter  $\rho(k)$ . In this chapter, two different scenarios will be considered: the first one

assumes no information about  $\rho(k)$ ; the second one considers that each  $\rho(k)$  is an independent and identically distributed process for which the probability distribution is known. Unless specifically mentioned, we are dealing with the broader case of no information.

When considering a polytopic uncertainty in the parameter  $\rho$ , the state equation in (7.1) becomes

$$x(k+1) = \left( A_0 + \sum_{\ell=1}^{n_\Delta} \Delta_\ell(k) A_\ell \right) x(k) + B(k)u(k) + L(k)d(k)$$

where  $n_\Delta$  is the number of required uncertainties and each  $\Delta_\ell(k)$  is a scalar uncertainty with  $|\Delta_\ell(k)| \leq 1$ . We assume for the sake of simplicity that matrices  $B(k)$ ,  $L(k)$ , and  $C(k)$  are parameter-independent and known, which can be relaxed by employing the techniques described in [Ros11] and [RS13]. In this chapter, we will assume two different scenarios where we have no information about  $\Delta(k)$  or when each  $\Delta(k)$  is an independent and identically distributed process to which we know the probability distribution. When not mentioned, we are dealing with the broader case of no information. The two following problems are addressed in this chapter.

**Problem 2** (Triggering in the worst-case). *Use the Set-Valued Observers (SVOs) framework to specify event- and self-triggered measurements when parameter  $\Delta(\cdot)$  has an unknown distribution.*

**Problem 3** (Triggering with stochastic information). *Use the Stochastic Set-Valued Observers (SSVOs) framework to specify event- and self-triggered measurements when the probability distribution  $p_\ell$  for each matrix  $A_\ell$  is known.*

In a distributed system or a networked control system conform with the description given by (7.1), matrix  $A(\rho(k))$  represents the dynamics which depends on the occurrence of a transmission. The observer might not be able to determine if the transmission was successful or which nodes communicated, thus leading to parameter  $\rho(k)$  being unknown. Matrix  $C(k)$  is either going to determine the sensors that are making a measurement update or be zero when in between updates.

To address Problem 2, we construct a set including all possible state realizations and obtain a bound on the error (i.e., the size of the computed polytope), the Set-Valued Observers (SVOs) described in Chapter 4 and Chapter 5 are adopted for general LPV systems.

In the context of Problem 3, the technique described in Chapter 2 corresponds to computing

$$\bar{X}(k+1) = \text{co} \left( \bigcup_{\theta_i \in \Theta} \text{Set}(M_{\theta_i}(k+1), m_{\theta_i}(k+1)) \right)$$

where  $\Theta$  is a smaller collection of the vertices of  $\mathcal{H}$  such that the probability of the state being contained in  $\bar{X}(k+1)$  is greater than or equal to  $1 - \alpha$ , and is therefore referred to as an  $\alpha$ -confidence set.

In the context of networked control systems, the problem being addressed can be summarized as how to use SVOs to determine event- and self-triggered strategies to determine when the sensors are required to perform a measurement. The method should provide a current



set-valued estimate for the state that should also characterize the maximum estimation error; defined as the greatest distance between the center and any point of the polytope.

For fault detection in a randomized distributed system, the objective is to reduce the amount of computations while ensuring that the set-valued estimates do not diverge. This is an important issue since the complexity grows exponentially both with the number of uncertainties (which depends on the number of possible transmissions) and also with the horizon  $N$ , since in (4.10),  $\mathcal{H}$  is of size  $2^{Nn_\Delta}$ . This problem can render the SVOs inapplicable for some systems, especially those with stringent time constraints such as in real-time control applications.

In the remainder of this chapter, it will be discussed how to equip the SVOs with event- and self-triggered strategies to reduce the network resources requirements, and also to reduce the complexity of the computations to a level where the state of time-sensitive applications can still be estimated resorting to an SVO-based technique.

## 7.4 Set-valued Estimate Approximations

### 7.4.1 Hyper-parallelepiped Approximation

The method to compute the set-valued state estimates makes use of polytopes and produces approximations to the *optimal* SVO which are always of the form  $\tilde{X}(k) = \{q : M(k)q \leq m(k)\}$ . Without loss of generality, one can redefine these sets to be of the form  $\tilde{X}(k) = \{q : M(k)q \leq 1\}$  assuming the origin is contained in the polytope. If this is not the case, one can simply shift the states so that the origin lies within the set.

The computation of unions and intersections of polytopes increases the number of vertices, which is a major limitation arising from the use of polytopes. A possible solution is to approximate  $\tilde{X}(k)$  by a polytope with bounded number of vertices and obtain a set  $\tilde{X}(k+1)$  that is more conservative than by simply considering, for instance, the convex hull. The additional error can be reduced by increasing the horizon  $N$ , as discussed previously. One possibility is to consider a hyper-parallelepiped overbound [Ros11] corresponding to the solution of the following linear program for each of the coordinate axis  $i$

$$\begin{aligned} s_{2i-1} = & \underset{x}{\text{minimize}} && e_i^\top x \\ & \text{subject to} && M(k+1)x \leq 1, \end{aligned} \quad (7.2)$$

to get the minimum of this linear combination, while the corresponding maximum is obtained

$$\begin{aligned} s_{2i} = & \underset{x}{\text{minimize}} && -e_i^\top x \\ & \text{subject to} && M(k+1)x \leq 1, \end{aligned} \quad (7.3)$$

which generates the polytope  $\tilde{X}(k+1) = \text{Set}(I \otimes \begin{bmatrix} 1 \\ -1 \end{bmatrix}, s)$  and can be put into the format where  $m(k) = 1$  by dividing each of the rows of  $I \otimes \begin{bmatrix} 1 \\ -1 \end{bmatrix}$  by the corresponding entry in vector  $s$ . Notice

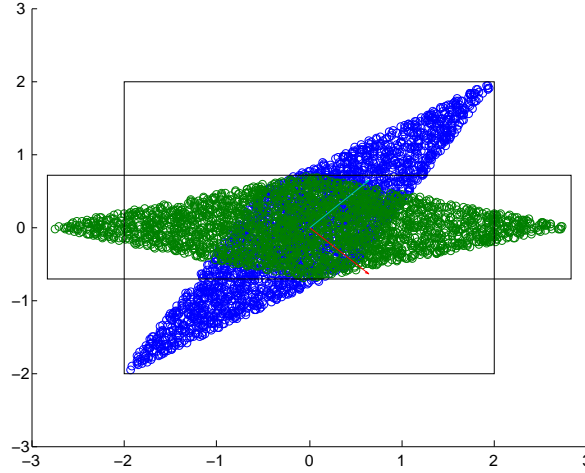


Figure 7.2: Original and rotated sets, blue and green respectively, and its correspondent overbounds.

that  $m(k) > 0$  from the assumption that the origin is contained in the polytope, which can be obtained, in turn, by performing the required translation.

The hyper-parallellepiped approximation performance depends on the structure of the given polytope. As an example, consider the set provided in blue in Figure 7.2, where the over-approximation would be a square of side 4. The area of the initial polytope is 8, whereas the area of its approximations is 16. If the set is “stretched” to get it closer to the line described by  $y = x$ , i.e., by increasing the condition number defined as

$$\kappa(M) = \frac{\sigma_{\max}(M)}{\sigma_{\min}(M)}$$

the overbound gets worse. In the following proposition, it is shown that the ratio between the hyper-volume of the set and the correspondent hyper-parallellepiped overbound can be arbitrarily large.

**Proposition 7.1.** *Consider the hyper-parallellepiped approximation defined in (7.2) and (7.3) and a polytope  $X = \text{Set}(M, 1)$ .*

$$\text{Then, } \exists M : \lim_{\kappa(M) \rightarrow \infty} \frac{\text{Vol}(M')}{\text{Vol}(M)} = \infty$$

where  $\text{Vol} : \mathbb{R}^{n \times m} \rightarrow \mathbb{R}$  maps a matrix  $\bar{M}$  into the hyper-volume of  $\text{Set}(\bar{M}, 1)$  and  $X' = \text{Set}(M', 1)$  is the hyper-parallellepiped approximation of  $X$  computed using (7.2) and (7.3).

*Proof.* Take the polytope with matrix  $M$  given by

$$M = \begin{bmatrix} -\frac{1}{\epsilon} & \frac{1}{\epsilon} & 0 & \dots & 0 \\ \frac{1}{\epsilon} & -\frac{1}{\epsilon} & 0 & \dots & 0 \\ \hline & & I \otimes \bar{1} & & \end{bmatrix}$$

where solving (7.2) and (7.3) returns  $M' = I \otimes \bar{1}$  for the hyper-parallellepiped  $X'$ . When  $\epsilon \rightarrow 0$  we have  $\kappa(M) \rightarrow \infty$  as  $\sigma_{\max}(M) \rightarrow \infty$ . We then have  $\text{Vol}(M') = 2^{n_x}$  while  $\lim_{\kappa(M) \rightarrow \infty} \text{Vol}(M) = 0$  which concludes the proof.  $\square$

The key observation to improve the accuracy of the SVO calculation is to rotate the set in blue in Figure 7.2 to align it with the coordinate axes (getting the set in green), obtaining in this way a less conservative overbound (in the example, the volume goes from 16 to 8). The depicted vectors represent the singular vectors of matrix  $M$  and define the directions of principal components of the sets. Therefore, the relationship with the condition number is clear in the sense that the higher the condition number, the more one direction is less predominant when compared to the others. In the extreme case of a  $\kappa(M) = \infty$ , one can conclude that the set has zero length in one of the dimensions.

The solution proposed in this section to find an improved overbound is a rotation to get the singular vectors as the canonical basis, as demonstrated in Figure 7.2. From the definition of the Singular Value Decomposition (SVD),  $M = USV^\top$  where the right-singular vectors are the columns of  $V$ , which are orthonormal, and the singular values are the elements of the diagonal of matrix  $S$ . Matrix  $M$  can be seen as the set after the rotation of the canonical vectors to match its singular vectors, i.e., the original set is rotated with respect to the canonical basis as depicted in blue in Figure 7.2. Then, to find the set with vectors aligned with the canonical basis depicted in green in Figure 7.2, i.e., the set defined by  $M_{\text{rot}}$ , we can write the relationship between  $M$  and  $M_{\text{rot}}$  by the rotation matrix  $R$  as

$$M_{\text{rot}} = (RM^\top)^\top = MR^\top. \quad (7.4)$$

Matrix  $R$  can be obtained through the equation

$$RV = I \Leftrightarrow R = V^\top \quad (7.5)$$

as we want to rotate from the singular vectors in  $V$  to the canonical vectors. By combining (7.4) and (7.5) we get

$$\begin{aligned} M_{\text{rot}} &= (RM^\top)^\top \\ &= (V^\top(USV^\top)^\top)^\top \\ &= USV^\top V \\ &= US. \end{aligned}$$

Thus, a possible approach to reduce the conservatism of a hyper-parallelepiped approximation is to apply a rotation using the singular vectors. In doing so, the principal axes of the set are aligned with the canonical vectors and the resulting hyper-parallelepiped overbound is tighter.

We can now address the conservatism issue of the approximation for a general polytope and perform a similar analysis to Proposition 7.1 but after applying the rotation to the polytope. The next proposition shows that the ratio does not depend on the condition number and has a factor depending solely on the state dimension.

**Proposition 7.2.** Consider a polytope  $X = \text{Set}(M, 1)$ , where the singular vectors of  $M$  are the canonical vectors after applying the rotation defined in (7.5). Take the hyper-parallelepiped approximation  $X' = \text{Set}(M', 1)$  of  $X$ , as defined in (7.2).

$$\text{Then, } \max_M \frac{\text{Vol}(M')}{\text{Vol}(M)} = n_x!$$

where  $\text{Vol}$  is defined as in Proposition 7.1.

*Proof.* We start by noticing that after the rotation, each of the hyper-faces of  $X'$  must contain at least a vertex of  $X$  which means that the worst case is to select the polytope  $X$  such that it has the lowest volume and at least a vertex in each of the hyper-faces of  $X'$ . This corresponds to select as  $X$  the  $n_x$ -simplex sharing  $n_x$  converging edges with the hyper-parallelepiped. In such case, we have

$$\text{Vol}(M) = \frac{\text{Vol}(M')}{n_x!}$$

which concludes the proof.  $\square$

It should be noticed that for a general polytope  $X$ , the proposed rotation is not desirable in all cases. If we select a counterexample as in Figure 7.3, the new set leads to a more conservative overbound. Nevertheless, the case in Figure 7.3 is caused due to the lack of *central symmetry* of the polytope as in Definition 7.1. If the polytope is made centrally symmetric, then the proposed rotation ensures that the hyper-parallelepiped overbound has at most a factor of  $n_x!$  increase in the hyper-volume of the set. Without using the rotation, the overbound can be arbitrarily large depending on the condition number of the matrix defining the polytope, as seen in Proposition 7.1.

**Definition 7.1.** A polytope  $X := \text{Set}(M, 1)$  is centrally symmetric if it can be written as the intersections of  $2\ell$  half-planes symmetric in pairs in relation to the origin, i.e. if  $M$  satisfies

$$M = \begin{bmatrix} m_1(k) \\ -m_1(k) \\ \vdots \\ m_\ell(k) \\ -m_\ell(k) \end{bmatrix}.$$

We now introduce an algorithm to make a polytope centrally symmetric, ensuring that the hyper-parallelepiped approximation after the rotation is not worse than the one before the rotation.

Algorithm 5 converts any general polytope in a centrally symmetric polytope with the important feature that the produced overbound does not increase the size of the hyper-polytopical and ellipsoidal techniques. The evolution of the set in different stages of the algorithm is illustrated in Figure 7.4.

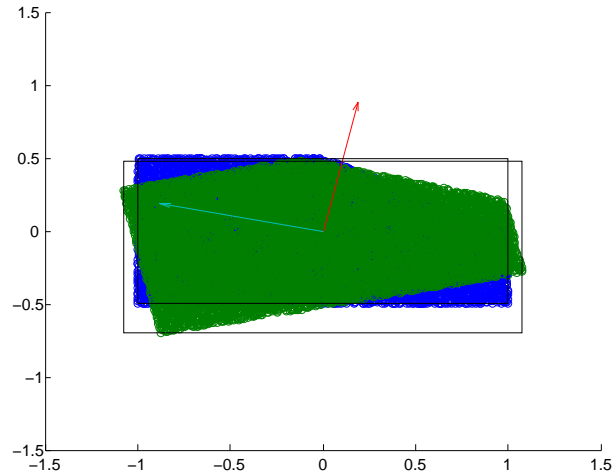


Figure 7.3: Counterexample where a set is rotated but a worst overbound is achieved.

---

**Algorithm 5** Centrally symmetric polytopes
 

---

**Require:** Polytope  $X := \text{Set}(M, 1)$ .

**Ensure:** Returns a polytope  $X$  which is centrally symmetric.

```

1: /* Center initial hyper-parallelepiped */
2: find s using (7.2) and (7.3)
3: apply translation to center hyper-parallelepiped defined by s
4: /* Add and remove rows */
5: for each row  $i$  do
6:   /* Test if intersects */
7:   if intersects( $X, -m_i$ ) then
8:     remove( $X, m_i$ )
9:   else
10:    add( $X, -m_i$ )
11:   end if
12: end for
13: return  $X$ 

```

---

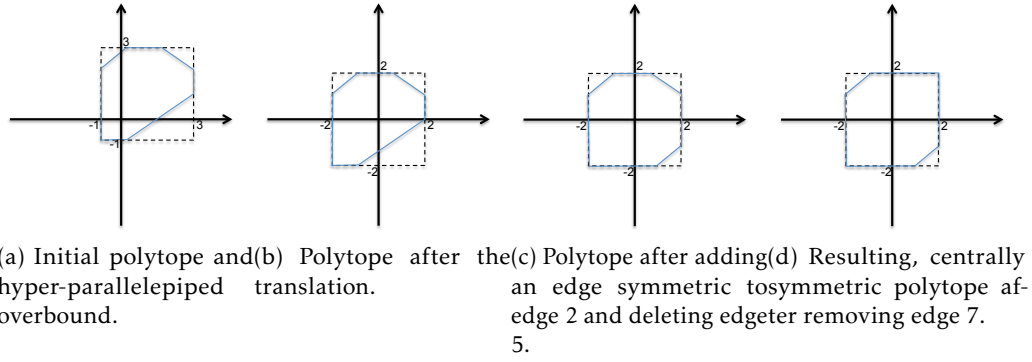


Figure 7.4: Example of the evolution of Algorithm 5 for a polytope that is not centered and not centrally symmetric. Edges are counted starting at the top one and counterclockwise.

### 7.4.2 Ellipsoidal Overbounding

The previous section introduced a rotation to deal with ill-conditioned cases in which a hyper-paralleliped overbound degrades performance. By reducing the conservatism of the overbound, the SVO design can relax the need for a large horizon to ensure convergence. This section aims to overbound the set-valued estimates by an ellipsoid, with the ultimate objective of having an easy-to-compute estimate, in case the accuracy can be temporarily reduced in order to improve computational performance.

The main limitation of SVO-based techniques when applied to a real-time or time-sensitive application is the associated computational burden. In each iteration, generating the set containing all possible state realizations amounts to the union of the sets obtained by propagating all possible combinations of the system dynamics and intersecting it with the set of states compatible with the current measurements. This process may be time-consuming, especially when the model of the system is only partially known - see [SRC<sup>+</sup>13].

In the next theorem, it is shown how an ellipsoidal overbound can be computed for a generic polytope that satisfies the centrally symmetric condition of Definition 7.1.

**Theorem 7.1.** *Consider a convex set*

$$\mathcal{S} = \{x : Mx \leq 1\}$$

*such that  $M \in \mathbb{R}^{n \times m}$  is as in Definition 7.1. An ellipsoidal overbound to  $\mathcal{S}$  is given by*

$$x^\top Q x \leq 1,$$

*where  $Q = \frac{VS^\top SV^\top}{n}$ .*

*Proof.* The inequality  $Mx \leq 1_n$  follows from the assumption that matrix  $M$  is as in Definition 7.1. We can infer  $\forall x : Mx \leq 1_n \Rightarrow x^\top M^\top Mx \leq 1_n^\top 1_n$ . After a singular value decomposition on  $M$ , the inequality becomes  $\frac{1}{n} x^\top VS^\top U^\top USV^\top \leq 1$  and, since  $U$  is an orthogonal matrix, we get the conclusion.  $\square$

A corollary of the previous result can be derived in order to provide an overbound in terms of the maximum norm of any point belonging to the set.

**Corollary 7.1.** *Consider a convex set  $\mathcal{S}$  such that matrix  $M$  is as in Definition 7.1. Then, an overbound to  $\mathcal{S}$  can be described by*

$$x^\top x \leq \frac{n}{\sigma_{\min}^2(M)}$$

which means that  $\|x\| \leq \frac{\sqrt{n}}{\sigma_{\min}(M)}$ .

*Proof.* Given the result in Theorem 7.1, then  $\forall x \in \mathcal{S} : x^\top Q x \leq 1$  with  $Q = \frac{V S^\top S V^\top}{n}$ . Then,  $x^\top Q' x \leq x^\top Q x$  with  $Q' = \frac{\sigma_{\min}^2}{n} V I V^\top = \frac{\sigma_{\min}^2}{n} I$  which yields the result.  $\square$

**Remark 7.1.** It should be stressed that alternative methods have been developed in the literature to obtain ellipsoidal set-valued state estimates, since the seminal work [Sch68], as described, for instance, in [Sch73]. However, the algorithm proposed in this section has some relevant properties, as discussed in the sequel, including the low-computational power required, as well as guaranteeing that the state of the system is indeed contained within the ellipsoid.

Recovering the abstract example in (4.9), the hyper-parallelepiped approximation would simply be the set described by the matrix  $M$

$$M = \begin{bmatrix} 0 & \frac{6}{5} \\ 0 & \frac{1}{5} \\ 10 & 0 \\ -10 & 0 \end{bmatrix}$$

and the ellipsoidal set would be given by matrix  $Q$  given by

$$Q = \begin{bmatrix} 50 & 0 \\ 0 & \frac{18}{25} \end{bmatrix}$$

which is depicted in Figure 7.5 where for this abstract system the set  $X(1)$  was not a particular bad choice as no rotation was needed.

## 7.5 Set-Valued Observer for Event- and Self-Triggered Systems

The framework of SVOs deals with a worst-case scenario and provides set-valued estimates where the state of the system is guaranteed to belong in contrast with providing a single estimate and a bound of the error for that estimate. In this section, we explore how to use the produced sets to define event conditions that, up to a certain extent, generalize those surveyed in Section 7.1 with the clear benefit of enabling other shapes for the barrier condition for triggering a sensor measurement.

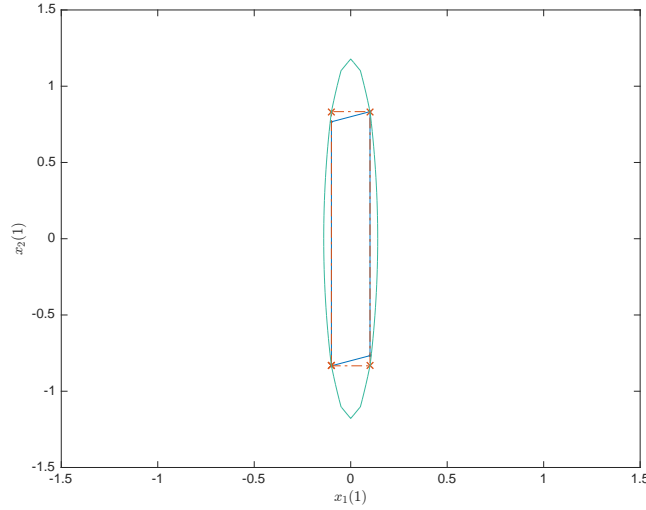


Figure 7.5: Abstract example where the previous set  $X(1)$  is enveloped by the hyper-parallelepiped approximation in dashed line and the ellipse upper bound.

### 7.5.1 Set-Valued Observers for Event-Triggered Systems

Event-triggered systems aim to reduce the communication burden between the sensors and the observer, which in networked control systems makes use of the shared medium network, thus consuming resources that may be critical to the remaining processes using the network. In the literature, the event trigger condition is common to be defined at the expense of the error of the last sensor update or as a quadratic or norm function of the state [HSB08]. However, the set-valued estimates of the state can also be used to provide an *event condition* for the sensor to perform a measurement.

An SVO constructs the polytopic approximation set  $\tilde{X}(k)$ , at each time instant  $k$ , for a system described by (7.1). The objective is to use this information and find an event condition such that the sensor can determine when a measurement update is required. We introduce the notation  $\tau^{-1}(k)$  to denote the last triggering time that is smaller than  $k$ . Similarly,  $\tau^1(k)$  refers to the first occurrence of a trigger that is greater than  $k$  and  $\tau^0(k) = k$  ( $\tau^0(k)$  will be used instead of  $k$  whenever we want to state explicitly that the current  $k$  is a triggering time). The second most recent trigger is denoted by  $\tau^{-2}(k) = \tau^{-1}(\tau^{-1}(k))$  and similarly for any other trigger.

A naive approach would entail the observer to send the matrix  $M(\tau^{-1}(k))$  and the vector  $m(\tau^{-1}(k))$  at time  $\tau^{-1}(k)$  to the sensor, which assuming knowledge of the full state, then tests if

$$x(k) \in \tilde{X}(\tau^{-1}(k))$$

and, if the sensor has a partial observation, it can check the more general condition

$$M(\tau^{-1}(k))C(k)^\dagger y(k) \leq m(\tau^{-1}(k)) \quad (7.6)$$

where the symbol  $\dagger$  stands for the Moore-Penrose pseudoinverse. For all subsequent time instants  $k+1, k+2, \dots$  the sensor needs to update matrix  $M(\tau^{-1}(k))$ , resorting to the nominal



dynamics (the matrix  $A_0$ ) and the control law  $B(k)u(k)$ . Condition (7.6) would easily not hold for cases where the stabilizing input signal has large magnitude. It is assumed that the sensor has access to the control signals from the controller as it is communicating to the plant. The update corresponds to compute  $M(\tau^{-1}(k))A_0^{-1}$  and apply the translation given by  $B(k)u(k)$ . When (7.6) does not hold using the last updated set, the sensor performs a measurement update and sends it to the observer that computes and sends back  $\tilde{X}(\tau^0(k))$ , which is the set for the current time.

The condition proposed in (7.6) can be viewed as a generalization of a condition depending on some norm. In particular, inequalities involving both the  $\ell_\infty$  norm, defined as  $\|x\|_\infty = \max_i |x_i|$ , and the  $\ell_1$  norm, defined as  $\|x\|_1 = \sum_{i=1}^n |x_i|$ , are polytopes and can be represented in this framework. In addition, the observer/controller can place additional constraints to trigger the update.

*Example:* Let us assume that the state of the system is a stock or other financial product quote and the observer/controller is a hedge fund manager running a supervisory system that triggers purchases and sells according to the received quotes. Due to regulation in the market, or motivated by correlation between products or even when having options and futures to cover the risk of other products, it might be useful to add new constraints to the transmitted condition. Such a condition cannot be represented using the previous norms. However, by using polytopes, extra linear restrictions can be represented by adding rows to  $M(\tau^{-1}(k))$ .

The conservatism of the initial set  $X(0)$  (by assumption  $\tilde{X}(0) := X(0)$ ) depends on the information available to the designer of the SVO. If little is known about the initial conditions of the systems, set  $X(0)$  must be made sufficiently large so as to contain any possible initial state  $x(0)$ . Condition (7.6) would be meaningless in this case, as sensor readings would not be triggered. We introduce a performance parameter  $\mu$  referring to the maximum allowed radius of the ball enclosing the polytope. Whenever

$$\frac{n_x}{\sigma_{\min}(M(k))} \geq \mu, \quad (7.7)$$

the observer requests a sensor measurement. The effect of  $\mu$  is to enforce the observer to construct a small set before setting an event condition.

The algorithm is summarized in Algorithm 6, where we use the notation  $\neg$  as the logical negation.

In NCSs, where the use of the network is an extremely valuable resource, one can opt by reducing the communication of the event condition by applying any of the overbounding techniques described in this chapter. If a hyper-parallelepiped or an ellipsoid approximation is used, the communication is reduced to the rotation matrix  $V$  of Theorem 7.1 multiplied by the expansion factors. Note that other techniques to reduce the size of transmitted information can be employed based on the exponential representation of the rotation matrix. If overbounding by a ball, the event condition resorts to an  $\ell_2$ -norm and only the radius is required for the sensor to determine when to trigger a measurement.

---

**Algorithm 6** SVOs for Event-Triggered systems

---

**Require:** Polytope  $X(0)$ .

**Ensure:** Event-triggered sensor updates.

```

1: for each  $k$  do
2:   if  $\neg(7.6)$  then
3:      $sensor\_update()$ 
4:      $\tilde{X}(\tau^0(k)) = svo\_update()$ 
5:     if (7.7) then
6:       /* Force a trigger by sending an empty set instead of  $\tilde{X}(\tau^0(k))$  */
7:        $send(empty\_set)$ 
8:     else
9:        $send(\tilde{X}(\tau^0(k)))$ 
10:    end if
11:  end if
12: end for

```

---

### 7.5.2 Set-Valued Observers for Self-Triggered Systems

Self-triggered systems require the ability to propagate estimates into future time steps, so as to determine the next sensor reading. The SVOs have the capability of forward propagation to get the time instant for the next sensor measurement, provided that the volume of the set-valued estimates does not grow beyond a certain, predefined limit.

Inequality (5.3) defined the estimation set for the state in the next time instant using the knowledge of the sensor measurement  $y(k)$ . However, removing the rows corresponding to the intersection with the measurements, defines the set-valued estimates that results only from propagating the dynamics, which we denote by  $\tilde{X}_p(k)$ . At time  $\tau^{-1}(k)$ , the observer receives the measurement  $y(\tau^{-1}(k))$ , and has access to the set  $\tilde{X}(\tau^{-2}(k))$ . To determine the next sensor update, a node resorts to (5.3) to find  $\tilde{X}(\tau^{-1}(k))$  and then propagates it using the dynamics in (7.1) to obtain  $\tilde{X}_p(\tau^1(k))$ , with  $\tau^1(k)$  being the first time instant such that

$$\tilde{X}_p(\tau^1(k)) \subseteq \tilde{X}(\tau^{-2}(k)). \quad (7.8)$$

The condition assures that the observation set does not increase in size because of the self-triggered approach.

For the above approach, finding  $\tau^1(k)$  can be performed by a logarithmic search, testing different values for  $\tau^1(k)$  as the size of  $\tilde{X}_p(k)$  is monotonically increasing, unless we have the stringent condition that the singular values of any chain of dynamics matrix are smaller than 1. To account for the more general case, we select  $\tilde{X}(\tau^{-2}(k))$  instead of  $\tilde{X}(\tau^{-1}(k))$  in (7.8). The procedure is summarized in Algorithm 7.

In systems where the computational power used in each time instant is limited, one can adopt a different strategy and have an iterative solution. By definition, we have the relationship  $\tilde{X}(k) \subseteq \tilde{X}_p(k) \cap Y(k)$ , since  $\tilde{X}(k)$  is the convex hull of the intersection between  $\tilde{X}_p(k)$  and  $Y(k)$ . To

---

**Algorithm 7** SVOs for Self-Triggered systems
 

---

**Require:** Polytope  $X(0)$ .

**Ensure:** Self-triggered sensor updates.

```

1: for each  $\tau^{-1}(k)$  do
2:   sensor_update()
3:    $\tilde{X}(\tau^{-1}(k)) = svo\_update()$ 
4:   find  $\tau^1(k)$  such that (7.8) is satisfied
5:   send( $\tau^1(k)$ )
6: end for
    
```

---

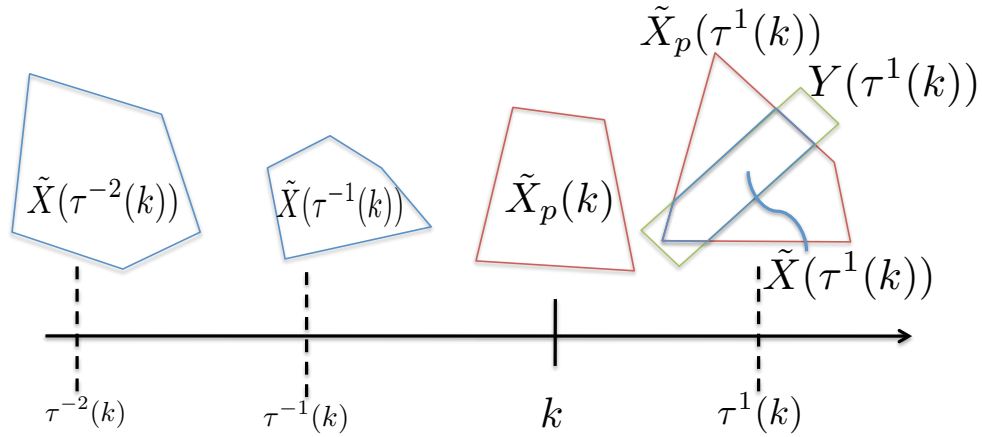


Figure 7.6: Example of using SVOs for self-triggered systems. At time  $\tau^{-1}(k)$ , the observer computes set  $\tilde{X}(\tau^{-1}(k))$  and propagates twice to get  $\tilde{X}_p(k)$  and  $\tilde{X}_p(\tau^1(k))$ , which is larger than  $\tilde{X}(\tau^{-2}(k))$ , and triggers a sensor measurement, making the intersection with the measurement set  $Y(\tau^1(k))$  to get the new estimation  $\tilde{X}(\tau^1(k))$ .

determine the set-valued estimate at time instant  $\tau^{-1}(k)$ , it is sufficient to compute  $\tilde{X}_p(\tau^{-1}(k)) \cap Y(\tau^{-1}(k))$ , which is an inexpensive computation. Instead of computing the set  $\tilde{X}(\tau^1(k))$  for different values of future times, one can resort to pre-computed products of matrices using the values for the uncertainties. Then, proceed to check if each of the products exceeds  $\tilde{X}(\tau^{-2}(k))$  in size, which is less computationally demanding since we prevented the computation of the convex hulls for all uncertainties for all values of next trigger time to be tested. The set  $\tilde{X}_p(\tau^1(k))$  can be computed during the inactivity time between  $\tau^{-1}(k)$  and  $\tau^1(k)$ , leaving  $\tilde{X}(\tau^1(k))$  to be computed at time  $\tau^1(k)$  from  $\tilde{X}_p(\tau^1(k))$ .

Figure 7.6 depicts how the sets in Algorithm 7 evolve with time. We draw attention to the fact that sets  $\tilde{X}_p(k)$  are monotonically increasing in volume, which motivated the triggering condition to use the set at time instant  $\tau^{-2}(k)$ .

## 7.6 Event- and Self-Triggered Set-Valued Observers

In the prequel, SVOs were used to determine only the triggering of sensors update (i.e., when the sensor needs to send a measurement to the observer) in the context of NCSs where only the number of updates is minimized in a greedy approach. Nevertheless, the SVOs estimates are

computed in every time instant, which motivates the introduction of Event- and Self-Triggered SVOs. The main objective is the reduction of the computational cost associated with the *classical* SVOs by only computing the set-valued state estimate (using the previously described tools) when this set is growing past the size of previous time instants. As an alternative, we compute and propagate overbounds, which are less computationally demanding, and perform the intersection with the measurement set for the worst-case in terms of system dynamics. The methods described for reducing the complexity of the SVO computations are compatible with the previous use of SVOs for event- and self-triggered systems.

The main advantage of this method is its real-time application due to diminished computational costs associated with three main factors:

- The matrix defining the polytope generally belongs to  $\mathbb{R}^{\ell \times n_x}$ , where  $\ell \gg n_x$  and represents the number of restrictions associated with the edges of the polytope whereas the proposed overbound matrix belongs to  $\mathbb{R}^{n_x \times n_x}$ ;
- Running the SVO computations only at a few time instants allows use of idle moments to pre-compute the necessary combinations of matrices products;
- In some special cases of interest, such as in distributed systems, it is possible to discard dynamics matrices based on the observation set and compute the worst-case estimate with minimal processing effort.

The first rows of (5.2) are equivalent to

$$M(k)(A_0 + A_{\Delta^*})^{-1} \mathbf{x} \leq m(k) \quad (7.9)$$

where (7.9) defines the set of points created by propagating the previous set defined by  $\text{Set}(M(k), m(k))$ . If the sets are always defined so as to have  $m(k) = 1_{n_x}$ , the state can be bounded at time  $k$  using Theorem 7.1. After computing the ellipsoidal overbound, it is propagated using the dynamics of the system, considering each given instantiation of the uncertainties in  $\Delta^*$ . Thus,  $X(k+1) = \{q : q = (A_0 + A_{\Delta^*})^{-1} x \wedge x \in X(k)\}$ .

From the previous discussion, the set  $\tilde{X}(k+1)$  can be described as

$$\tilde{X}(k+1) = \text{co}\left(\bigcup_{\Delta \in \mathcal{H}} \text{Set}(M(k)(A_0 + A_{\Delta}(k))^{-1}, 1)\right)$$

If the original set defined by the matrix  $M(k)$  is overbounded using the procedure found in Section 7.4.2, the resulting overbound set  $\tilde{X}(k+1)$  corresponds to the convex hull of the union of ellipsoids, where their axis correspond to the singular vectors basis (see the illustration presented in Figure 7.7).

Due to sensor noise and/or the inability to measure the full state of the system, the observations can be defined as a set  $Y(k)$  which is a polytope posing constraints on the current state.

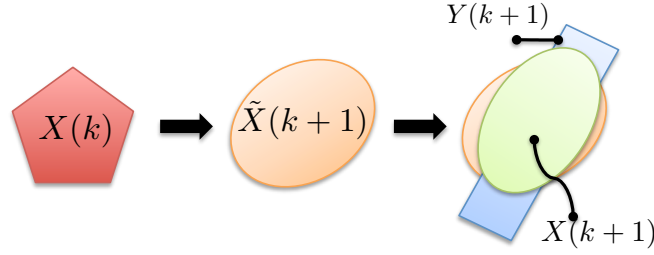


Figure 7.7: Original set and ellipsoidal overbound with the set resulting from the intersection with the measurement set to form the new set-valued estimate.

The singular values and the associated singular vectors of the matrix defining such a polytope indicate the directions where the uncertainty is greater. Therefore, to test whether to execute the operation of computing the actual set, one can resort to intersecting the observation set  $Y(k+1)$  with the ellipsoids and evaluate if the norm of the state increases over the current iteration to prevent it from becoming arbitrarily large. In other words, this allows us to derive conditions to decide whether to use the SVO procedure described in Section 4.4 or update the overbound.

An easy-to-compute new estimate for the norm of the state at time  $k+1$  can be obtained by solving

$$\begin{aligned} & \text{maximize} \quad \|p\| \\ & \text{subject to} \quad p^\top M(k+1)^\top M(k+1)p \leq 1 \\ & \quad \quad \quad p \in Y(k+1). \end{aligned}$$

The previous problem translates into finding an intersection of ellipsoids and then computing the point in that set with the greatest norm. Matlab's Ellipsoidal Toolbox [KV06] provides computationally efficient methods to tackle this problem. The complexity associated with computing the intersection of two sets is constant in terms of required iterations and each is cubic in the dimension of the state, as it amounts to solving a Second-Order Cone Programming (SOCP).

### 7.6.1 Event-Triggered Set-Valued Observers

The description of how an Event-Triggered SVO works is similar to how an event-triggered system performs the sensor updates, as seen in Section 7.5.1. An event condition is based on requiring that the approximation ellipsoid is contained in the current maximum norm ball of radius  $\mu(k)$ . The value  $\mu(k)$  is the minimum between a performance bound specified by the user and the last approximation set maximum norm, so as to guarantee convergence of estimates.

The Event-Triggered SVO computes an ellipsoid overbound that approximates the set  $\tilde{X}(k)$  but which is *fast* to compute and fairly inexpensive when compared to some of the required computations of a “classical” SVO, such as the convex hull of (5.2) for each of the uncertainty instantiations. A *full* iteration of the SVO is going to be triggered at time  $\tau^0(k)$  if the following

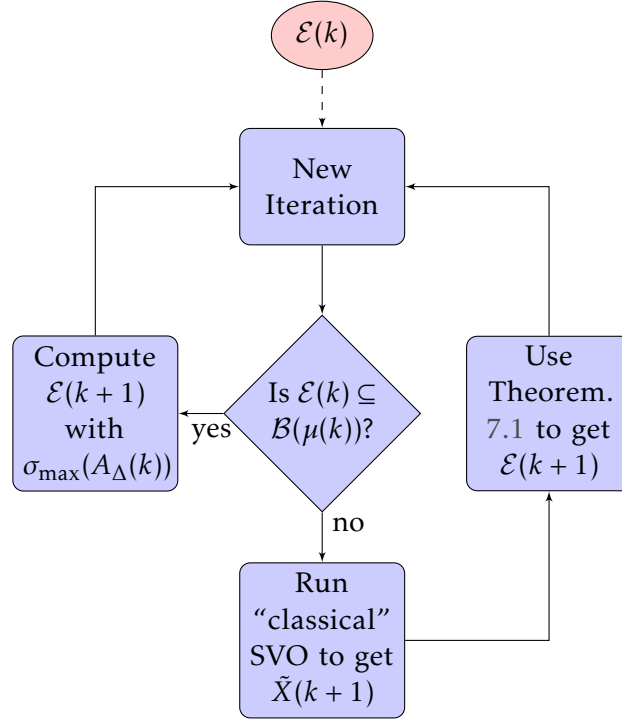


Figure 7.8: Flowchart of the Self-Trigger SVO algorithm where  $\mathcal{E}(k)$  and  $\mathcal{B}_{\mu(k)}$  are the overbounding ellipsoid at time  $k$  and the ball of radius  $\mu(k)$  centered at the origin, respectively.

does not hold

$$\mathcal{E}(\tau^0(k)) \subseteq \mathcal{B}(\mu(\tau^{-1}(k))) \quad (7.10)$$

where  $\mathcal{E}(\tau^0(k))$  is the ellipsoid approximation at the current time and  $\mathcal{B}(\mu(\tau^{-1}(k)))$  is a ball centered at the origin of radius  $\mu(\tau^{-1}(k))$ . When triggered, the observer gets  $\tilde{X}(\tau^0(k))$  using (5.2) and then using Theorem 7.1 we can obtain the new  $\mathcal{E}(\tau^0(k))$ . The new radius for the event condition is given by

$$\mu(\tau^0(k)) = \min(\mu_u, \frac{n_x}{\sigma_{\min}^2(M(\tau^{-1}(k)))}) \quad (7.11)$$

where  $\mu_u$  is a user provided performance minimum. Notice that it is not possible to use  $M(\tau^0(k))$  in (7.11) because if no measurement is available this would result in the mechanism triggering every instant. This algorithm is summarized in Figure 7.8.

The event condition used for the SVOs is very similar to that of Section 7.5.1 and enables the use of both strategies so as to avoid communications between the sensor and the observer. Additionally, the observer can return set-valued estimates without a heavy computational burden in between sensor updating times. In such a scenario, the SVO can output the sets  $\mathcal{E}(\tau^{-1}(k)), \mathcal{E}(\tau^{-1}(k)+1), \dots, \mathcal{E}(k)$ , until  $\mathcal{E}(k) \not\subseteq \mathcal{B}(\mu(\tau^{-1}(k)))$ . The set  $\mathcal{E}(k)$  increases in hyper-volume since there is no intersection with the measurement set. In summary, if both strategies were to be used together, the sensor would be testing whether its last observation is within the received set to trigger an event, whereas the observer is outputting the ellipsoidal sets for the worst-case until their hyper-volume is larger than that at the last trigger.

### 7.6.2 Self-Triggered Set-Valued Observers

Self-Triggered SVOs aim to have the important feature of allowing the node run by the observer to be shut down during the time between each trigger. Nonetheless, the set-valued estimates for that period might be needed by some application, which is somehow contradictory. However, the observer can take advantage of the computations performed when it was determining the next trigger instant, where it computed the propagated set for all time instants from the current time until the next trigger. These sets can be used as the set-valued estimates in between updates.

The procedure is identical to the triggering mechanism detailed in Section 7.5.2, but using the update condition in (7.10). In essence, at each trigger time, the SVO will run the standard SVO iteration and obtain the polytope representing the set-valued estimate, and then compute the next trigger time. An inexpensive solution is to find the ellipsoidal approximation and propagate it in successive iterations until condition (7.10) is no longer satisfied. Doing so avoids computing the polytope sets for each of the time instants in between triggers.

The search for the next trigger time produces the set-valued estimates that are necessary for all the remaining future time instants in a lightweight fashion. As a consequence, there is no computation between triggers. We note that event- and self-triggered strategies can be combined at two different levels. For example, the observer can be running a Self-Triggered SVO and, at each triggering time, providing the estimates up to the next triggering time, while the sensors might use that sequence of estimates to determine an event-triggered sensor update. The processes requiring the estimates can intersect the event-based sensor updates with the self-triggered estimates.

The main difference between Self-Triggered SVOs and SVOs used for self-triggered systems relies on where the computational effort lies. In the former, computational power is being saved on the observer side, by reducing the number of necessary state estimations using the standard SVO procedure. In the latter, the focus is on the network usage by the sensors in their updates.

The most advantageous combination is to have a Self-triggered SVO sending ellipsoidal approximations in between triggers to the sensor. Then, an Event-triggered strategy can be used at the sensor, testing whether the last received ellipsoid still contains the current measurement. In doing so, a communication only happens when the ellipsoid at the sensor does not include the current measurement. At this time instant, the sensor will send a batch of new measurements and will receive back a new ellipsoid as state estimate. On the other hand, the computational load is also reduced since a full SVO computation is triggered only when the current ellipsoid is not a suitable estimate. The full procedure will use the whole batch of measurements sent by the sensor since the last full computation produced all the ellipsoids that might be requested by the sensor in an event fashion (see Figure 7.9 for a visual depiction of the interaction between sensor and observer).

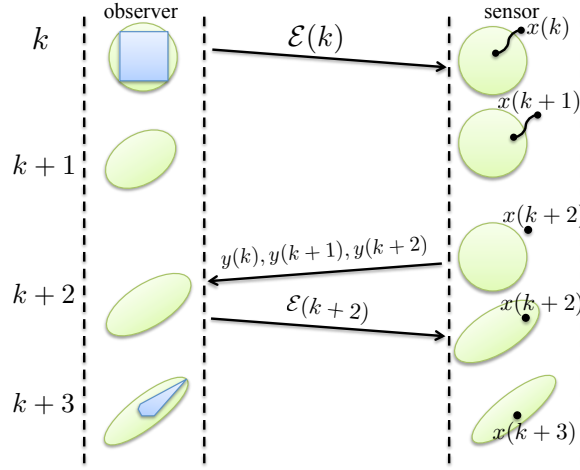


Figure 7.9: Depiction of the observer and sensor sets for a combination of a Self-Triggered SVO used with an event-triggered NCS.

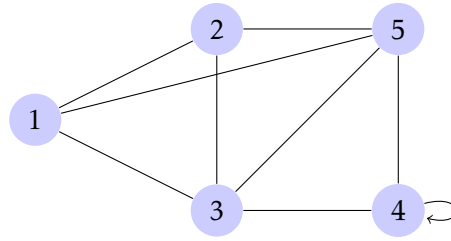


Figure 7.10: Network example for a distributed system.

### 7.6.3 Distributed Systems

The case of distributed systems is particularly relevant when considering fault detection applications. In particular, given the formulation in (7.1), it is possible to accommodate a distributed system by considering that each node is represented as a state and that the sequence of actions of each node defines the  $\Delta$  parameters that selects a given overall system dynamics at any time instant. In Figure 7.10, we depict an example of a network for a distributed system. Parameter  $\Delta$  can represent, for instance, the communication between two nodes, i.e. the realization of the edges of the graph [SRC<sup>+</sup>13].

The definition also encompasses the case of distributed gossip algorithms where node selection and communication times are random processes. In this subsection, the systems to be considered satisfy the assumption that all dynamics matrices are equal apart from a reordering of the rows and columns. The case of gossip algorithms fits this description in the sense that at each time instant a random pair of nodes performs some given operation using their states (see [MR10]). Such systems motivated the analysis of systems with the referred assumption which we formally introduce in the following definition

**Definition 7.2** (reordering property). *A dynamic system as in (7.1) has a reordering property in*



its nodes if the dynamics written as  $A_0 + A_{\Delta_j}$ ,  $\forall j : 1 < j \leq n_x$  satisfy

$$A_0 + A_{\Delta_j} = P(A_0 + A_{\Delta_1})P^\top, \forall j \quad (7.12)$$

where matrix  $P$  is a permutation matrix and  $j$  is any node different from 1.

**Remark 7.2.** Definition 7.2 can be found for example in the case of distributed gossip algorithms. However, more generally, one can have  $P$  being any orthogonal change of basis (i.e.,  $\forall j, A_0 + A_{\Delta_j} = P(A_0 + A_{\Delta_1})P^\top$  with  $P \in O(n)$ ).

Following Definition 7.2, let us introduce the following proposition which means that any ellipsoid set resulting from the propagation of the dynamics are the same up to a permutation change of basis.

**Proposition 7.3.** Take any ball  $B_b := \{q : \|q\| \leq b\}$  and matrices  $A_0 + A_{\Delta_j}$ ,  $1 \leq j \leq n_x$  as in Definition 7.2.

Then, all ellipsoids  $\mathcal{E}_j := \{q : \frac{1}{b^2} x^\top (A_0 + A_{\Delta_j})^{-\top} (A_0 + A_{\Delta_j})^{-1} x \leq 1\}$  are equal to  $\mathcal{E}_1$  up to a rotation.

*Proof.* From (7.12) and the fact that the permutation matrix is orthogonal (i.e.,  $P^\top P = I$ ), we get

$$\forall i : \sigma_i(A_0 + A_{\Delta_j}) = \sigma_i(A_0 + A_{\Delta_1}), \quad (7.13)$$

where  $\sigma_i(A)$  is the  $i$ th singular value. We also have

$$\frac{1}{b^2} x^\top (A_0 + A_{\Delta_j})^{-\top} (A_0 + A_{\Delta_j})^{-1} x = \frac{1}{b^2} (P^\top x)^\top (A_0 + A_{\Delta_1})^{-\top} P^\top P (A_0 + A_{\Delta_1})^{-1} (P^\top x)$$

and due to (7.13)

$$\sigma\left(\frac{1}{b^2} (A_0 + A_{\Delta_1})^{-\top} P^\top P (A_0 + A_{\Delta_1})^{-1}\right) = \sigma\left(\frac{1}{b^2} (A_0 + A_{\Delta_1})^{-\top} (A_0 + A_{\Delta_1})^{-1}\right)$$

with matrix  $P$  defining a rotation. Thus, the conclusion follows.  $\square$

Proposition 7.3 allows the introduction of the following theorem that limits the number of computation to verify if the norm passed the condition event to that of testing if the singular vectors of each of the propagated ellipsoids aligns with the singular vectors of  $Y(k)$ .

**Theorem 7.2.** Consider a distributed algorithm with  $\tilde{X}(0) = \{p : \|p\| \leq 1\}$ , each dynamics matrix being written as  $A_0 + A_{\Delta_j} = U_j S V_j^\top$ , with  $v_{\max}^j$  being the singular vector associated to the largest singular value and, conversely, an observation set  $Y = \{U_y S_y V_y^\top p : \|p\|_\infty \leq 1\}$  with  $v_{\max}^y$  corresponding to the largest singular value. Then, the worst-case set-valued state estimate overbound is given by the intersection of  $Y$  with the ellipsoid defined by  $\Delta_j$  such that

$$\max_j (v_{\max}^j)^\top v_{\max}^y$$

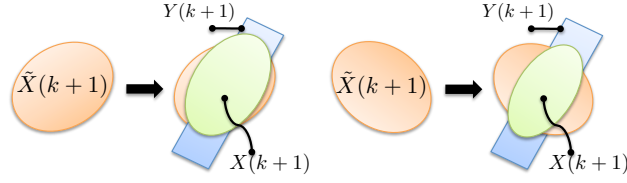


Figure 7.11: Example demonstrating two ellipsoids and its corresponding intersection with the set of observations.

*Proof.* From Proposition 7.3 we get that all the ellipsoids computed using each of the parameters  $\Delta_j$  are equal up to a rotation. Finding the largest intersection between any of these ellipsoids and  $Y$  can be found as the one for which the following optimization program has the highest solution:

$$\begin{aligned} & \text{maximize} \quad \|p\| \\ & \text{subject to} \quad p^\top M_j p \leq 1 \\ & \quad \quad \quad p \in Y. \end{aligned}$$

where the  $M_j = U_j S^{-2} V_j^\top$ , following the conversion to ellipsoid representation. Since all matrices  $M_j$  are the same apart from a rotation, we are solving the equivalent problem

$$\begin{aligned} & \text{maximize} \quad \|p\| \\ & \text{subject to} \quad (R^\top p)^\top M_1 (R^\top p) \leq 1 \\ & \quad \quad \quad p \in Y \\ & \quad \quad \quad R \in \{R_1, \dots, R_j\}. \end{aligned}$$

When  $R$  is unconstrained, has a closed-form solution given by  $RV_y = V_1$  and the cost function evaluates to

$$\min(\sigma_{\max}(S_y), \sigma_{\max}(A_0 + A_{\Delta_1}))$$

i.e., the maximum singular vectors align. The optimization goal  $\|p\|$  is monotonically increasing with the inner product of the maximum singular vectors of the ellipsoids and measurement set  $Y$ . Thus, the constrained version of the problem has a solution for the ellipsoid with the maximum inner product between its singular vectors and the singular vector of  $Y$  and the conclusion follows.  $\square$

Theorem 7.2 establishes that the triggering condition presented in this chapter is particularly effective in the context of distributed systems having the reordering property. In such systems, the worst case scenario can be found by checking the inner product between the maximum singular vector of each possible dynamics matrix and the singular vectors of the observation set. This fact is illustrated in Figure 7.11 where two ellipsoidal overapproximation sets are shown with the corresponding intersection with the set  $Y$ . As a consequence, at each time instant, the mechanism selects the overbounding ellipsoid producing the largest intersection. By doing so, the computational cost associated with the combinatorial behavior of the SVOs becomes

constant given that only one ellipsoid needs to be computed along with one intersection and no unions are needed. This is irrespective of the number of dynamics matrices (i.e., the number of agents and states in the network).

## 7.7 Triggering Frequency and Convergence

Self-triggered systems have the advantage of reducing the communication associated with the process at a frequency that depends on the characteristics of the system and its sensors. In this section, we analyze the triggering frequency by showing how the time until the next update can be inferred from the singular values of the system dynamics. We also demonstrate that the triggering techniques do not prevent estimate convergence of the standard SVO. For Event-triggered systems, such analysis cannot be performed as the trigger depends on the actual measurement value, but the results for Self-triggered can be viewed as a worst-case for the Event-triggered, in the sense that the condition for triggering corresponds to obtaining a measurement that is the *worst* possible, from the point of view of the stability of the set-valued estimates.

An important issue when introducing such a technique is its impact on the convergence of estimates. We refer Proposition 4.2 proved in [Ros11] regarding the boundedness of the produced sets in terms of hypervolume, also presented in Chapter 4. Based on the condition in Proposition 4.2, we can introduce the counterpart for the convergence with the triggering schemes.

**Proposition 7.4.** *Suppose that a system described by (7.1) satisfies Proposition 4.2 for a given  $N^*$ . Then, the following conditions are satisfied:*

- i) *A Self-Triggered SVO cannot grow without bound by considering  $N \geq N^*$ ;*
- ii) *An Self-Triggering System using an SVO has estimates that cannot grow without bound by considering  $N \geq N^*$ .*

Proposition 7.4 comes directly from the fact that no assumptions are required in Proposition 4.2 regarding the measurements. In both cases there is, at some point in time, a computation of the standard set-valued estimates using SVOs. Proposition 4.2 takes into consideration only the dynamics of the system. The next theorem presents a similar result incorporating the effect of the intersection with the measurement set. The intuition behind the result is that in the directions that the system is measured, the requirement for stability can be dropped, as the intersection will decrease the uncertainty in those directions.

**Theorem 7.3 (SVO convergence).** *Suppose that a system described by (7.1) with  $x(0) \in X(0)$  and  $u(k) = 0, \forall k$ , verifies, for sufficiently large  $N^*$ ,*

$$\gamma_N := \max_{\Delta(k), \dots, \Delta(k+N)} \left\| \text{null}(C(k+N)) \prod_{j=k}^{k+N} \left[ A_0 + \sum_{i=1}^{n_\Delta} \Delta_i(j) A_i \right] \right\| < 1 - \delta_N,$$

## Chapter 7: Event- and Self-Triggered strategies

---

for all  $N \geq N^*$ , where  $\text{null}(C(k+N))$  is the matrix defining a null space orthonormal basis of  $C(k+N)$  and

$$\delta_N := \max_{d(k), \dots, d(k+N-1)} \|A_k^{N-1} L(k) d(k) + \dots + L(k+N-1) d(k+N-1)\|.$$

Then, the hypervolume of  $\tilde{X}(k)$  is bounded.

*Proof.* Consider the ellipsoidal overbound given by Theorem 7.1 for the state at time  $k$ , denoted by  $\mathcal{E}(k)$ , where without loss of generality  $\mathcal{E}(k) = \{x : \|x\| \leq 1\}$ . The maximum norm of any point belonging to any  $\mathcal{E}(k+N)$  satisfies

$$\begin{aligned} \|x(k+N)\| &\leq \gamma_N \|x(k)\| + \delta_N \\ &\leq \gamma_N + \delta_N \end{aligned}$$

since the intersection along each of the directions in  $C(k+N)$  is at most  $2\nu^*$  as it is the size of  $Y(k+N)$ .

If  $N \geq N^*$ ,  $\|x(k+N)\| \leq 1$  implying  $\mathcal{E}(k+N) \in \mathcal{E}(k)$  and additionally  $\tilde{X}(k+N) \in \mathcal{E}(k)$  and the conclusion follows.  $\square$

Theorem 7.3 refines the result in Proposition 4.2 by noticing two facts: we can discard the directions associated with the measurement matrix  $C(\cdot)$  since the maximum size of the intersection with  $Y(k)$  is going to be  $2\nu^*$ ; and, in the worst-case, the decrease in norm associated with the dynamics compensates the increase associated with the disturbance signal.

### 7.7.1 Worst-case Scenario

In this subsection, results regarding Problem 2 are presented. The next theorem gives an “easy-to-compute” alternative to the iterative testing of different triggering times introduced in this chapter. Intuitively, the result shows how the size of the set-valued estimates relates to the system dynamics and allows use of an (off-line) pre-computed sub-optimal value for the triggering time.

**Theorem 7.4.** Consider a Self-Triggered SVO, as in Section 7.6.2, with maximum state norm at the trigger time  $\tau^{-2}(k)$  given by  $\|x(\tau^{-2}(k))\| \leq \mu(\tau^{-2}(k))$ . The next trigger  $\tau^1(k)$  occurs after  $\mathcal{T}_k := \tau^1(k) - k$  time instants, where

$$\mathcal{T}_k = \left\lceil \log_{\gamma} \sigma_{\min}(M(k)) \frac{\mu(\tau^{-2}(k))(1-\gamma) - \sqrt{n_d}}{\sqrt{n_x}(1-\gamma) - \sqrt{n_d}\sigma_{\min}(M(k))} \right\rceil$$

with

$$\gamma = \max_{i \in \{1, \dots, n_{\Delta}\}} \sigma_{\max}(A_0 + A_{\Delta_i})$$

*Proof.* Any point  $x_1 \in X(k+1)$  satisfies  $x_1 = (A_0 + A_{\Delta})x_0$  for some  $x_0 \in X(k)$  and some realization of the uncertainties  $\Delta$ . Since it is assumed that the self-triggering technique translates, at each

time instant, the set to incorporate the control law  $B(k)u(k)$ , then,

$$\begin{aligned} \|x(k + T_k)\| &= \|(A_0 + A_{\Delta_{k+T_k}})(A_0 + A_{\Delta_{k+T_k-1}}) \cdots (A_0 + A_{\Delta_k})x(k) \\ &\quad + (A_0 + A_{\Delta_{k+T_k-1}}) \cdots (A_0 + A_{\Delta_k})L(k)d(k) + \cdots + L(k + T_k)d(k + T_k)\| \\ &\leq \gamma^{T_k} \|x(k)\| + \sum_{j=0}^{T_k-1} \gamma^j \sqrt{n_d} \\ &\leq \gamma^{T_k} \frac{\sqrt{n}}{\sigma_{\min}(M(k))} + \sqrt{n_d} \frac{1 - \gamma^{T_k}}{1 - \gamma} \end{aligned}$$

which holds for the non-trivial case of  $\gamma \neq 1$ . To maintain the norm bounded  $\|x(k + T_k)\| \leq \mu(\tau^{-2}(k))$  it is required that

$$\begin{aligned} \gamma^{T_k} \left( \frac{\sqrt{n_x}}{\sigma_{\min}(M(k))} - \frac{\sqrt{n_d}}{1 - \gamma} \right) &\leq \mu(\tau^{-2}(k)) - \frac{\sqrt{n_d}}{1 - \gamma} \\ \Leftrightarrow \\ \gamma^{T_k} &\leq \frac{\mu(\tau^{-2}(k)) - \frac{\sqrt{n_d}}{1 - \gamma}}{\frac{\sqrt{n_x}}{\sigma_{\min}(M(k))} - \frac{\sqrt{n_d}}{1 - \gamma}} \\ \Leftrightarrow \\ T_k &\leq \log_{\gamma} \sigma_{\min}(M(k)) \frac{\mu(\tau^{-2}(k))(1 - \gamma) - \sqrt{n_d}}{\sqrt{n_x}(1 - \gamma) - \sqrt{n_d}\sigma_{\min}(M(k))} \end{aligned}$$

thus, leading to the conclusion.  $\square$

Theorem 7.4 presented a relationship between the system dynamics and the triggering frequency. An analogous result can be derived for the case of a system where the dynamics are selected from a set of possible matrices according to a stochastic variable. In such a setup, the probability distribution for the parameter  $\rho$  is known. Practical examples of this model range over distributed stochastic systems where some decision is random or the nodes acting at a given time instant are stochastically chosen.

### 7.7.2 Stochastic case

Another case of interest is to analyze the triggering frequency when the probability distribution for the uncertain parameter  $\rho$  of matrix  $A(\rho(k))$  is known (i.e., in the context of Problem 3). Before stating the result, the following definitions are required, where  $\inf$ ,  $\sup$  and  $\emptyset$  denote respectively the infimum, the supremum and the empty set.

**Definition 7.3** (volume expansion stochastic variable). *For a distributed system where the dynamics is selected from a set  $\{A_0 + A_{\Delta_i} : 1 \leq i \leq n_{\Delta}\}$  following a probability distribution where  $A_{\Delta_i}$  is selected with probability  $p_i$ , define the sequence of volume expansion stochastic variable as  $\theta(k) = \sigma_{\max}(A_0 + A_{\Delta_i(k)})$ , with probability  $p_i$ .*

Notice that the stochastic variable for the volume expansion is the stochastic equivalent of the quantity  $\gamma$  in Theorem 7.4. The results in this section only assume the knowledge of the

expected value of the distribution and not the distribution itself, since we are focusing on the expected value for the triggering frequency. The interested reader is referred to [WQF15] for results that require prior knowledge of the probability distribution.

**Definition 7.4** (upcrossing). *For a sequence of stochastic variables  $Z_1, \dots, Z_n$  and two real numbers  $a$  and  $b$ , define*

$$S_{k+1}(Z) = \inf\{n \geq T_k(Z) : Z_n \leq a\} \text{ and } T_{k+1}(Z) = \inf\{n \geq S_{k+1}(Z) : Z_n \geq b\}$$

*with the usual convention that  $\inf \emptyset = \infty$ . The number of upcrossings of the sequence  $Z$  of the interval  $[a, b]$  in  $n$  time instants is defined as*

$$U_n([a, b], Z) = \sup\{k \geq 0 : T_k(Z) \leq n\}.$$

Notice that the definition of upcrossing in Definition 7.4 of random variables is going to be equivalent to a trigger in our application. The volume exceeding the triggering condition is represented by the random variable corresponding to that volume making an upcrossing of the interval. We now introduce the theorem stating the results for the triggering frequency in randomized algorithms, where  $\mathbb{E}$  denotes the expected value operator and  $\mathbb{P}$  the probability function.

**Theorem 7.5.** *Consider a distributed system and a stochastic variable as in Definition 7.3. Then,*

- i) if  $\mathbb{E}[\theta(n)] < 1$ , the volume of the set-valued estimates converges almost surely to a nonnegative integrable limit and  $\mathbb{P}[\text{"having a trigger"}] \leq \frac{\mu(k)}{\mu(\tau^{-2}(k))}$ ;*
- ii) if  $\mathbb{E}[\theta(n)] = 1$ , the expected triggering time is given by  $\mathbb{E}[T_k|k] = \log \mu(k) \log \frac{\mu(\tau^{-2}(k))}{\mu(k)}$ , where  $T_k := \tau^1(k) - k$ ;*
- iii) if  $\mathbb{E}[\theta(n)] > 1$ , the time before the expected value of the number of triggers is greater than or equal to 1 is given by the  $M$  that satisfies  $\mathbb{E}[|Z_M - Z_0|] < \mu(\tau^{-2}(k)) - \mu(\tau^{-1}(k))$ .*

*Proof.* i) Consider the stochastic process  $Z_{n+1} = Z_n \theta(n)$ ,  $Z_0 = \mu(\tau^{-1}(k))$  describing the behavior of the size of the set-valued estimates for the distributed system. Also consider the correspondent filtration  $\mathcal{F}_n = \{Z_0, Z_1, \dots, Z_n\}$  (for additional information on martingale theory, see [Wil91]).

Computing the conditional expectation, we get

$$\begin{aligned} \mathbb{E}[Z_{n+1}|\mathcal{F}_n] &= \mathbb{E}[\theta(n)Z_n|\mathcal{F}_n] \\ &= Z_n \mathbb{E}[\theta(n)|\mathcal{F}_n] \\ &= Z_n \mathbb{E}[\theta(n)] \\ &< Z_n \end{aligned}$$

which implies  $Z_n$  is a nonnegative supermartingale and  $\lim_{n \rightarrow \infty} Z_n = Z_\infty$  with  $Z_\infty$  being a nonnegative integrable variable as stated in page 148 of [Pol02]. The final conclusion is a direct application of the Dubin's Inequality that states

$$\mathbb{P}[\text{"at least } \zeta \text{ upcrossings"}] \leq \left(\frac{a}{b}\right)^\zeta$$

where  $a$  and  $b$  define upcrossings as in Definition 7.4. In our context,  $a$  is the initial value and  $b$  represents the maximum volume before a trigger. Thus, the number of upcrossings is the number of triggers. As a consequence, if  $Z_n$  reaches 0 it must stay there forever.

ii) Consider the above martingale  $Z_{n+1} = Z_n \theta(n)$ ,  $Z_0 = \mu(k)$  and define the new martingale  $V_n = \log Z_n$ . Therefore, we have that  $V_{n+1} = V_n + \xi(n)$ , where  $\xi(n)$  is  $\log \theta(n)$ , along with the correspondent filtration  $\mathcal{F}_n = \{V_0, V_1, \dots, V_n\}$ .

Computing the conditional expectation, we get

$$\begin{aligned} \mathbb{E}[V_{n+1}|\mathcal{F}_n] &= \mathbb{E}[V_n + \xi|\mathcal{F}_n] \\ &= \mathbb{E}[V_n|\mathcal{F}_n] + \mathbb{E}[\xi|\mathcal{F}_n] \\ &= V_n + \mathbb{E}[\xi]. \end{aligned}$$

By definition,  $\mathbb{E}[\xi(n)] = \log \mathbb{E}[\theta(n)]$  which implies that  $\mathbb{E}[V_{n+1}|\mathcal{F}_n] = V_n$  and indeed  $V_n$  is a martingale. We progress by writing the stochastic variable  $W_n = V_n^2 - n$  and showing that it can indeed be made a martingale. Take the correspondent filtration  $\mathcal{F}_n = \{V_0, V_1, \dots, V_n\}$  and let us compute

$$\begin{aligned} \mathbb{E}[W_{n+1}|\mathcal{F}_n] &= \mathbb{E}[V_{n+1}^2 - (n+1)|\mathcal{F}_n] \\ &= \mathbb{E}[V_n^2 + 2V_n\xi + \xi^2 - (n+1)|\mathcal{F}_n] \\ &= V_n^2 + 0 + \mathbb{E}[\xi^2|\mathcal{F}_n] - (n+1). \end{aligned}$$

Without loss of generality, we assume variable  $\xi$  to have the expected value of its square equal to 1. This can be achieved by scaling the state of the system. Thus, simplifying to

$$\begin{aligned} \mathbb{E}[W_{n+1}|\mathcal{F}_n] &= V_n^2 - n \\ &= W_n \end{aligned}$$

Let us consider the stopping time corresponding to our self-triggering technique

$$\mathcal{T}_k = \inf \{n \geq 0 : V_n = \log \mu(\tau^{-2}(k))\}$$

Due to the martingale properties,  $V_{\mathcal{T}_k \wedge n}$  is a martingale which implies that

$$\mathbb{E}[V_{\mathcal{T}_k \wedge n}|k] = \mathbb{E}[V_{\mathcal{T}_k \wedge 0}|k] = \mathbb{E}[V_0|k] = \log \mu(k).$$

We can also compute the probability of hitting the maximum volume  $\log \mu(\tau^{-2}(k))$  by computing

$$\begin{aligned} \mathbb{E}[V_{\mathcal{T}_k}|k] &= \log \mu(\tau^{-2}(k)) \mathbb{P}[V_{\mathcal{T}_k} = \log \mu(\tau^{-2}(k))|k] \\ &\Leftrightarrow \\ \mathbb{E}[V_0|k] &= \log \mu(\tau^{-2}(k)) \mathbb{P}[V_{\mathcal{T}_k} = \log \mu(\tau^{-2}(k))|k] \\ &\Leftrightarrow \\ \mathbb{P}[V_{\mathcal{T}_k} = \log \mu(\tau^{-2}(k))|k] &= \frac{\log \mu(k)}{\log \mu(\tau^{-2}(k))}. \end{aligned}$$

Using the new martingale

$$\mathbb{E}[W_{\mathcal{T}_k \wedge n}|k] = \mathbb{E}[W_{\mathcal{T}_k \wedge 0}|k] = \mathbb{E}[W_0|k] = (\log \mu(k))^2, \quad (7.14)$$

and also

$$\mathbb{E}[W_{\mathcal{T}_k \wedge n}|k] = \mathbb{E}[V_{\mathcal{T}_k \wedge n}^2 - \mathcal{T}_k \wedge n|k] \quad (7.15)$$

Using both (7.14) and (7.15), we get

$$\mathbb{E}[V_{\mathcal{T}_k \wedge n}^2|k] = (\log \mu(k))^2 + \mathbb{E}[\mathcal{T}_k \wedge n|k]. \quad (7.16)$$

Due to the Monotone Convergence theorem, as  $\mathcal{T}_k \wedge n \rightarrow \mathcal{T}_k$ ,  $\mathbb{E}[\mathcal{T}_k \wedge n|k] \rightarrow \mathbb{E}[\mathcal{T}_k|k]$ , which combined with (7.16) leads to

$$\mathbb{E}[V_{\mathcal{T}_k}^2|k] = (\log \mu(k))^2 + \mathbb{E}[\mathcal{T}_k|k] \quad (7.17)$$

but by definition

$$\mathbb{E}[V_{\mathcal{T}_k}^2|k] = (\log \mu(\tau^{-2}(k)))^2 \mathbb{P}[V_{\mathcal{T}_k} = \log \mu(\tau^{-2}(k))|k] = (\log \mu(\tau^{-2}(k)))^2 \frac{\log \mu(k)}{\log \mu(\tau^{-2}(k))} \quad (7.18)$$

Using (7.17) and (7.18), we get that  $\mathbb{E}[\mathcal{T}_k|k] = \log \mu(k) \log \frac{\mu(\tau^{-2}(k))}{\mu(k)}$ , thus reaching the conclusion.

iii) We consider the submartingale  $Z_n$  and recall the Upcrossing Lemma that states

$$\mathbb{E}[U_M^{\alpha, \beta}] \leq \frac{\mathbb{E}[|Z_M - Z_0|]}{\beta - \alpha}$$

for a submartingale  $Z_n$ . To get  $\mathbb{E}[U_M^{\alpha, \beta}] < 1$ , we must get

$$\frac{\mathbb{E}[|Z_M - Z_0|]}{\beta - \alpha} < 1$$

which is satisfied by selecting  $M$  as in the statement of the theorem where  $\alpha$  is the current hypervolume of the set-valued estimates and  $\beta$  is the maximum allowed hypervolume and the conclusion follows.  $\square$

## 7.8 Simulation Results

In this section, we start by illustrating the advantages of the proposed event- and self-triggering techniques in order to reduce sensor updates. We consider a linearized model of the inverted pendulum mounted on a cart, which relates directly to the real-world example of an attitude control of a booster rocket at takeoff. In continuous time, the state dynamics are given by

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-(I+m\ell^2)b}{I(M+m)+Mm\ell^2} & \frac{m^2 g \ell^2}{I(M+m)+Mm\ell^2} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-m\ell b}{I(M+m)+Mm\ell^2} & \frac{mg\ell(M+m)}{I(M+m)+Mm\ell^2} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{I+m\ell^2}{I(M+m)+Mm\ell^2} \\ 0 \\ \frac{m\ell}{I(M+m)+Mm\ell^2} \end{bmatrix} u + Lw$$

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} + Nn$$



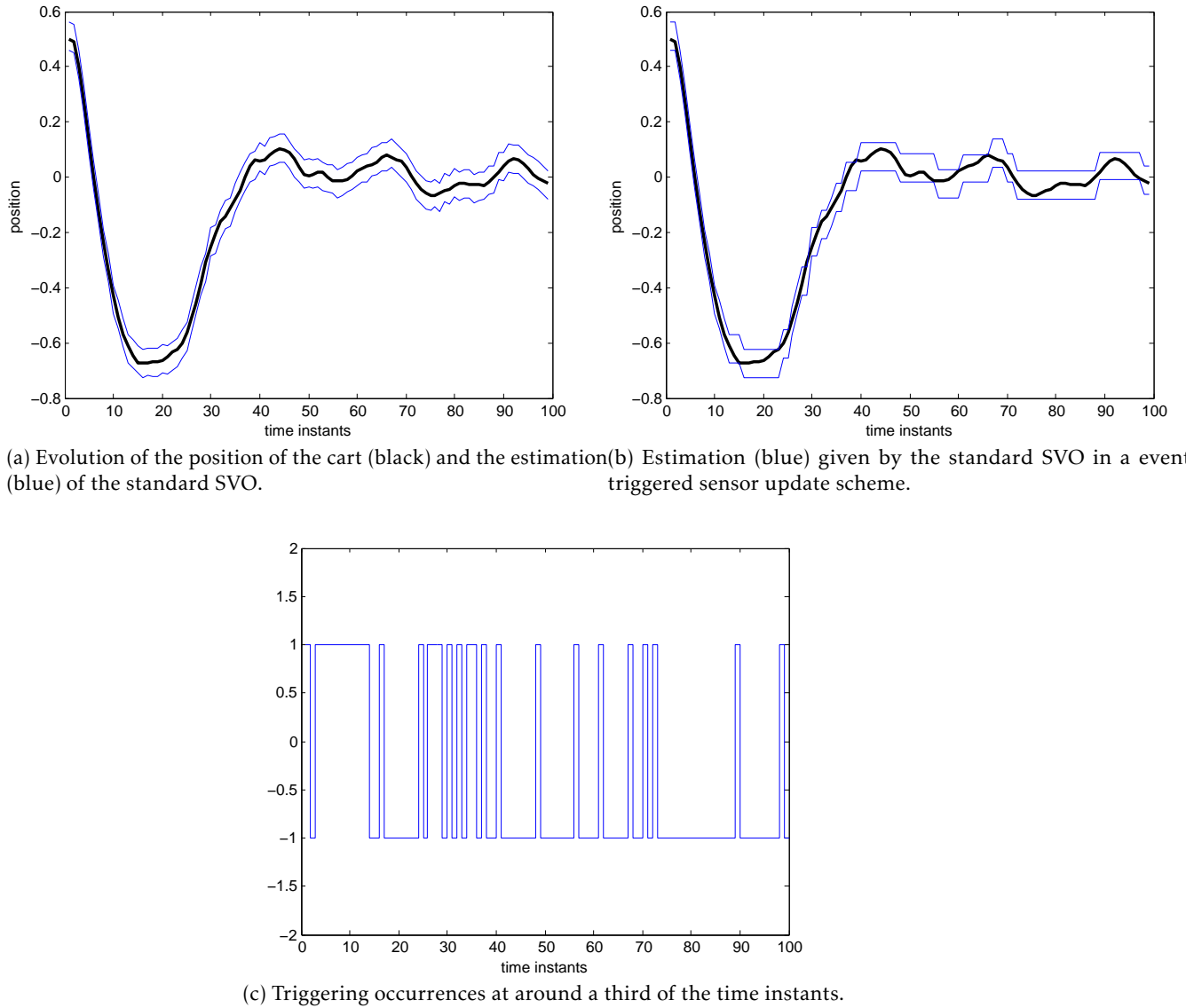


Figure 7.12: Estimation conservatism and triggering frequency of the event-triggering strategy for NCS using the standard SVOs.

where  $x$  is the cart position coordinate and  $\theta$  is the pendulum angle from vertical. The constants appearing in the model are the moment of inertia of the pendulum ( $I = 0.006 \text{ kg.m}^2$ ), length to pendulum center of mass ( $\ell = 0.3\text{m}$ ), coefficient of friction for the cart ( $b = 0.1 \text{ N/m/sec}$ ), mass of the pendulum ( $m = 0.2\text{kg}$ ), and mass of the cart ( $M = 0.5\text{kg}$ ). The system is discretized using a sampling period of  $0.1\text{s}$ .

We assume a matrix  $L$  for the disturbances equal to two and a half times the input matrix as to make the problem harder by having a large disturbance signal, and the noise injection matrix  $N$  is  $\begin{bmatrix} 0.5 & 0.5 \end{bmatrix}^T$ . The random signal  $w$  is taken from a normal distribution with variance equal to 1 and mean 0 with a maximum imposed of at most 5, which represents a large disturbance signal when compared to the control input. The control law  $u$  is assumed to be given by a

state-feedback controller, independent from the SVO, that returns a signal which stabilizes the unperturbed system. The objective is for the SVO in a NCS to provide estimates for the state of the remote system comprised of both the controller and the plant.

The first simulations focus on showing the properties of using SVOs to produce event conditions for the sensor to determine when to perform an update and send information through the network to the observer, i.e., SVOs for Event-triggered NCS as in Section 7.5.1. Figure 7.12 presents the main results of using an event condition based on the produced set-valued estimates of the state. In Figure 7.12b, it is depicted the interval for the values of the state that the observer outputs with the sensor updating according to the signal in Figure 7.12c as opposed to having sensors updates at every time instant, which would result in the estimates given in Figure 7.12a. The main observation is that the technique does not introduce conservatism in the estimates as the observer makes them constant within triggers and the current estimates are *validated* by the sensors which otherwise would trigger an update.

The event-triggered for NCS strategy simulated in Figure 7.12 showed that for the considered system, the triggering occurrences happen at around one third of the time instants. Such a reduction motivates our contribution of using SVOs to determine triggering strategies as considerable load on the networked would be avoided in comparison with the standard approach of receiving measurements in all the time instants.

The event-triggering strategy required the sensor unit to test whether the measurements are still inside the provided event condition set. We simulated the self-triggered version as to compare the results, i.e., SVOs for Self-triggered NCS as in Section 7.5.2. In Figure 7.13, it is depicted the same results for a different run of the algorithm but still allowing to point out the trade-off between both strategies. Contrarily to the event-triggered condition, it is observed in Figure 7.13b that the size of the estimation set changes due to some conservativeness being introduced by not having access to the sensor update. However, the convergence properties of the SVOs are maintained, since upon a trigger, the standard procedure is executed.

Figure 7.13c shows the occurrence of the triggers that corresponds to approximately 60% of the time instants having a trigger for running the standard SVO procedure. The main reason is the large disturbance and noise signals that make the produced sets grow in hyper-volume when no measurement is available. In essence, to have the possibility to switch off sensors in a self-triggered strategy, for this scenario, there is a twofold increase in the number of triggers and a poorer estimate quality when compared with the previous one. Nevertheless, the contribution of using SVOs to self-trigger NCSs should be seen for a different use when the sensor nodes are not equipped with relevant computational capabilities and all operations must be performed at the observer node. A saving of roughly  $\frac{1}{3}$  of the network resources associated with communication is still encouraging.

A third simulation is performed resorting to the same example but for the Self-triggered SVOs as in Section 7.6.2. In the previous cases, all computations were done using the traditional

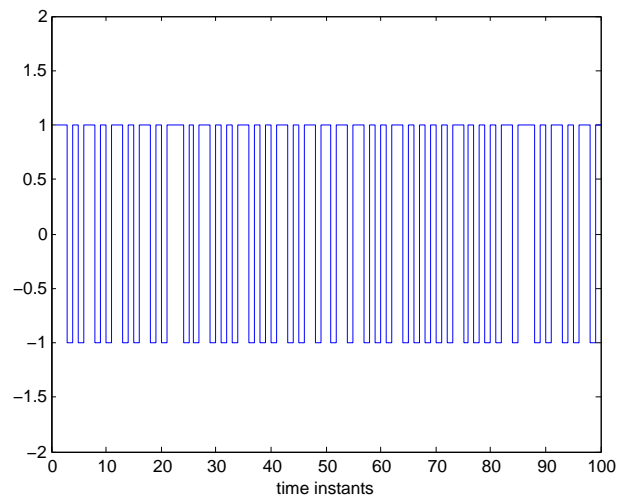
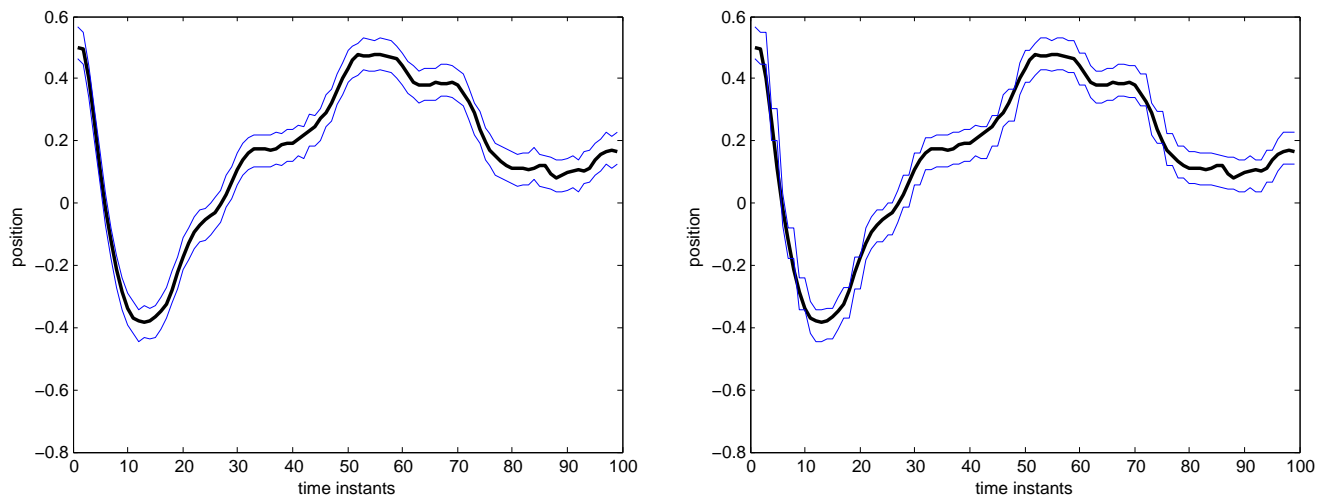


Figure 7.13: Estimation conservatism and triggering frequency of the self-triggering strategy for NCS using the standard SVOs.

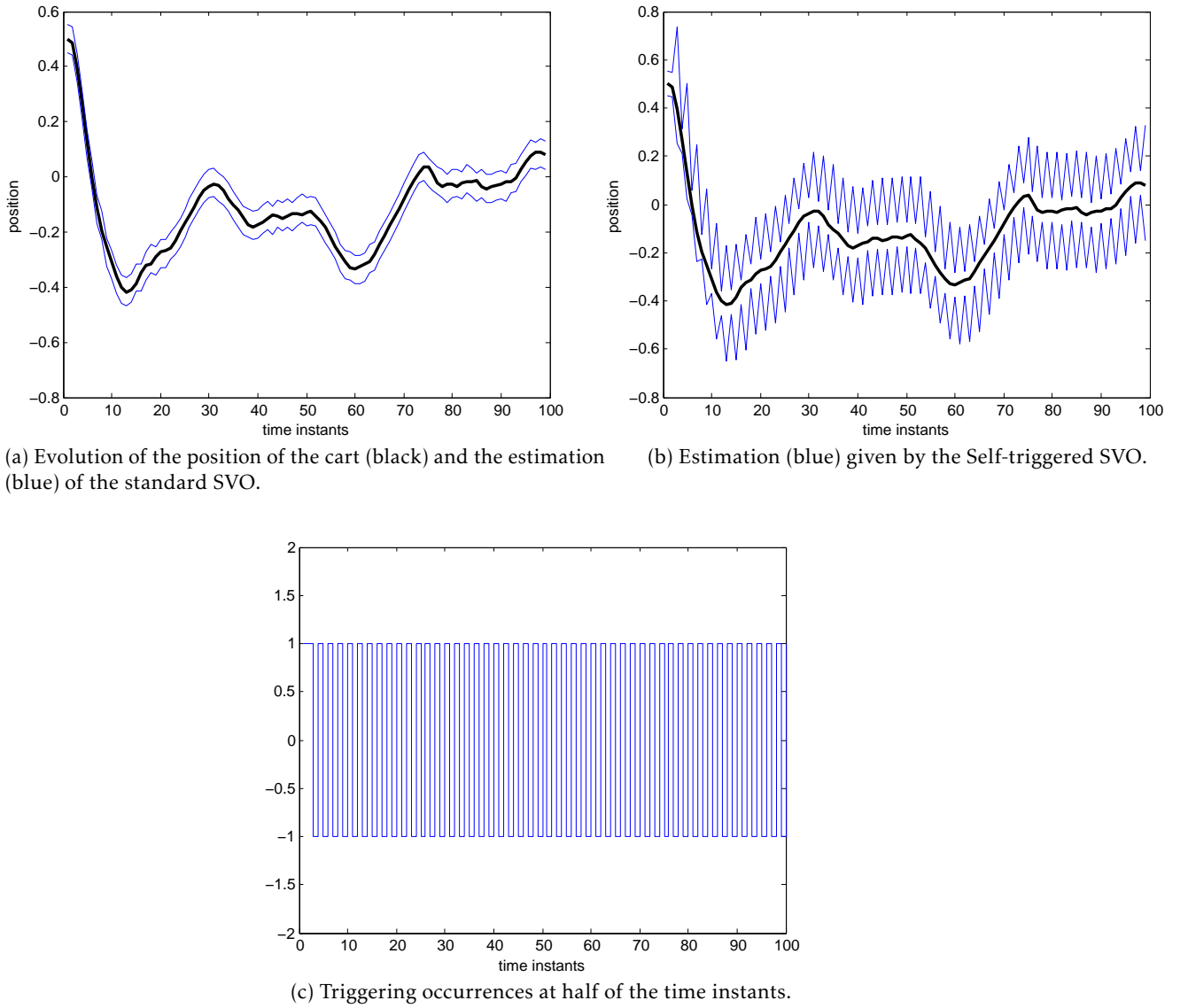


Figure 7.14: Estimation conservatism and triggering frequency of the Self-triggered SVOs in comparison with the standard SVOs.

SVOs whenever sensors updates were available. In this simulation, sensor updates are available at every time instant and triggers mean that the standard SVOs were computed.

The results are shown in Figure 7.14. For the run depicted in Figure 7.14a, we have the computed set-valued estimates in Figure 7.14b using the overbounding methods as aforementioned. A main difference is the introduced conservatism due to the ellipsoidal overbounding method, which is worsen by propagating for all possible values of the disturbance and noise signal. The triggering occurrences in this run were around 50% of the time instants as shown in Figure 7.14c. The main conclusion from this simulation is that the self-trigger SVO can be seen as an alternative to the traditional one especially in the cases where the disturbance and noise signals have a small magnitude. When that is not the case, this example gives evidence that the

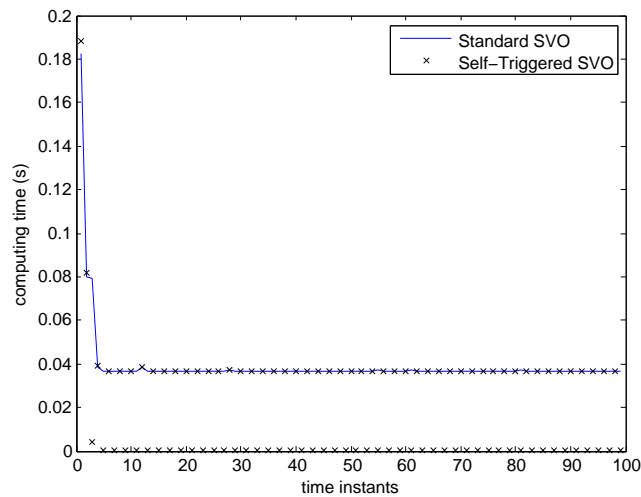


Figure 7.15: Elapsed time in seconds of the computation of the estimates using the standard and Self-triggered SVOs.

triggering frequency is high given that the set-valued estimates after a trigger are never of such a small volume as to avoid a considerable increase in overhead.

Nevertheless, a Self-Trigger SVO can expand the class of systems to which the SVOs can be applied. A special consideration is always systems that need to be discretized with small sampling periods or real-time plants. In those cases, time constraints are of utmost importance and place strict performance lower bounds on any potential technique. In Figure 7.15, the computing time for the traditional and Self-trigger SVOs is depicted. The minimum computational time for the traditional SVO was  $3.6 \times 10^{-2}$ s in contrast with  $1.6 \times 10^{-4}$ s for the proposed overbounding method, representing a decrease of two orders of magnitude. Thus, Self-triggered SVOs can be employed for system with stricter time constraints as long as the full computations at triggering times can be performed in between triggers, suggesting some future work in this topic.

## 7.9 Conclusions

In this chapter, the problem of reducing the network load in a NCS was addressed resorting to event- and self-triggered strategies. For this purpose, the concept of SVOs was used to provide polytopes where the state is known to belong and the triggering condition is selected such that the hyper-volume of the set-valued estimates does not grow. The algorithm does not impact on the convergence of the estimates since when the estimates increase in size, a sensor update is required to reduce the uncertainty in the estimates.

Following the study of triggering techniques, we provided similar event conditions to determine when to run a full computation of the SVOs to find the polytope for the estimates or compromise accuracy to gain in performance by giving low-complexity hyper-parallelepiped and ellipsoidal overapproximations. These were obtained by introducing an algorithm to find

a centrally symmetric polytope for the estimates. This was guaranteed to not deteriorate the approximations and by performing a rotation, the case of polytopes defined with ill-conditioned matrices, which can lead to arbitrarily bad approximations, was shown to introduce an error factor that at maximum grows with the factorial of the state space.

The convergence of the proposed strategies was shown as long as the conditions for the convergence of SVOs are satisfied by the system. In this chapter, it was presented a novel result for convergence that generalizes an existing result in the literature by noticing that along the directions associated with the rows of matrix  $C(\cdot)$ , the maximum size of the polytope is always at most  $2\nu^*$  and therefore is independent of the dynamics of the system.

The triggering frequency was studied and shown to depend on the maximum singular value of the possible dynamics matrix and the minimum singular value of the matrix defining the polytope, which *measures* the maximum norm of the previous estimation set. The case of distributed systems was also addressed when the probability distribution for the dynamics matrix is known for the cases where the expected value for the maximum singular value is smaller, equal to or greater than one.

The work presented in this document suggests a natural course for future developments. Two main avenues of research will be pursued: an extension of the described event- and self-triggering techniques to other set-based methods and what additional results can be provided for different set descriptions; and alternative optimization techniques that can be employed to determine the next self-triggered time instant apart from generating all the ellipsoids and checking whether they satisfy the triggering criteria.

# 8

## CONCLUSIONS AND FUTURE DIRECTIONS

This dissertation addressed several problems within the scope of fault detection and isolation in distributed system governed by stochastic selection of dynamics, alongside with developments of the SVO framework to consider many issues that were still open. The main goal was to create algorithms and develop tools that are distributed and enable Networked Control Systems (NCSs) to be robust to faults.

For time-dependent networks, a fault-tolerant algorithm was designed to deal with crash-type faults by introducing randomness in the nodes communication. It uses asynchronous updates and unidirectional messages, working both for the broadcast and gossip interaction. The thesis presents convergence results in stochastic sense and makes clear the connection between convergence rate in the continuous and discrete time domains. Exploiting the dependence of the expected value on the second largest eigenvalue of the probability matrix, a distributed optimization is carried out using common steps for addressing separable variables.

The assumption of the network evolution being independent is dropped and the case of social network is considered. A novel model is presented that aims at incorporating how people interact to form an objective opinion regarding a topic. Different state-dependent network dynamics are studied determining the impact on the convergence rate for the deterministic case. Finite-time rates and the contribution of each agent to the final opinion is provided. The results are meaningful for social media related topics and marketing campaigns, but also because they can directly translate to control applications, namely those involving vehicles with wireless communications.

Convergence in the presence of leaders or stubborn agents is another useful case which is tackled in this thesis before introducing randomness to account for the asynchrony in people interactions in the social context and to deal with network faults for the control version. In some cases, by appropriately selecting the network parameters it is possible to obtain the nodes converging to the average consensus which is somehow interesting for the applications considered in the time-dependent case.

Distributed systems can be modeled as uncertain Linear Parameter-Varying (LPV) systems where the parametric uncertainty translate the fact that not all nodes know which group of agents is communicating. By casting the problem as an LPV, a Set-Valued Observer (SVO) is designed to perform fault detection or isolation of multiple faults and make the algorithm tolerant to a broader class of faults, in the sense, that the impact of faults is limited and possible to compute beforehand.

The class of stochastic faults where the fault is a possible dynamics of the system but its probability distribution does not follow the one defined by the algorithm was also considered. Resorting to the definition of  $\alpha$ -confidence sets, the Stochastic Set-Valued Observers (SSVOs) are introduced to construct a set where the state is known to belong with probability  $1 - \alpha$  and following the concept of testing whether the current measurements can be given by dynamics that obey the model and its probabilities.

Building upon the use of SVOs and SSVOs for fault detection and isolation, an algorithm for average consensus sharing estimates is given that, in finite-time, either returns the final consensus value if no fault has occurred or detects it for some communication patterns. If that is not the case, asymptotic convergence of the algorithm to the consensus value is proved.

The computational complexity of the SVOs and its application to plants that lose observability due to having relative measurements is addressed by considering a left-coprime factorization of the system. In doing so, the dynamics of the two subsystems can be made arbitrarily fast for the observable LPV and dependent on the slowest unobservable mode for the detectable LPV case.

Real-time applications or plants discretized with a small sampling period require stringent constraints in the elapsed time taken by the SVOs to produce estimates. Event- and Self-triggering conditions are presented to temporarily compromise accuracy to reduce computational complexity. The proposed criteria does not prevent the convergence of estimates or the results for the standard case. Similarly, the SVOs are used to provide conditions to event- and self-triggered NCSs where the main goal is to reduce the network usage by having less frequent sensor updates.

The triggering frequency is addressed for the self-triggered case which also overbounds the event-triggered solution. In real applications, it translates in the observer having no *a priori* information about the variation of the parameters in the dynamics matrix. Focus is also given to the setting where the expected value of the maximum singular value for the dynamics matrix is known. In addition, the previous result requiring the system to be stable for having a bounded growth on the size of the estimates is generalized and shown that the system can indeed have unstable directions as long as they are compensated by the measurements.



## 8.1 Future Directions

Several issues related to the problems that we have addressed remain open. In particular, future research endeavors can be taken along the following directions:

- The SSVOs consider the stochastic information available regarding the dynamics matrix since the uncertainty is in which node communicated. The tool can be generalized to consider the probability distributions of various parameters and the signals in the system to cope with a broader class of problems. Motivation for research in this direction includes: computing the reliability metrics of not losing a file in a network where nodes are entering and leaving; stochastic communications in Sensor networks; computing the probability of losing a file in a Peer-to-Peer network; or a driver selecting a route and changing the congestions load in a Traffic Network;
- Current SVO definition already regards the parameters in its equations but its information is discarded since the estimation focuses on the state. A novel approach for performing sensitivity analysis can be researched in order to assess how system parameters influence the performance of the network. In particular, how the selection of device communication range, switching-off policies, number of deployed sensors, concentration distribution, etc affects a Sensor Networks lifetime and performance. Sensitivity analysis also plays a key role in other network analysis problems, e.g. how to reduce the number of parameters in a given network model, by finding the relevant ones;
- State estimation for state-dependent dynamics such as in the case of Social Networks. Studying how to include only the combinations of dynamics that abide to the state-dependent updating rule in the state estimation tools is of prime importance. Current SVOs deal with time-dependent parameters by considering the worst-case that inherently adds conservatism. The application of this family of tools would be manifold from the social networks, to nonlinear algorithms with conditions or Traffic Networks where each driver's decision impacts on the probabilities of other drivers picking a different route.



# A

## APPENDIX

For a Linear Time-Invariant (LTI) system of the form:

$$\begin{aligned}x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) + Du(k),\end{aligned}$$

where  $x(k) \in \mathbb{R}^n$ ,  $u(k) \in \mathbb{R}^d$  and  $y(k) \in \mathbb{R}^m$ , the canonical Kalman Decomposition allows to write the system in a new basis for the state such that the unobservable and uncontrollable modes are separated from the controllable and observable ones.

Let us define the similarity transformation  $T$  for the state

$$\bar{x} := T^{-1}x, \quad T := \begin{bmatrix} T_{c\bar{o}} & T_{co} & T_{\bar{c}\bar{o}} & T_{\bar{c}o} \end{bmatrix}.$$

where:

- the columns of  $T_{c\bar{o}}$  form a basis for the subspace  $\mathcal{C} \cap \mathcal{UO}$ ,
- the columns of  $\begin{bmatrix} T_{c\bar{o}} & T_{co} \end{bmatrix}$  form a basis for the controllable subspace  $\mathcal{C}$  of the pair  $(A, B)$ ,
- the columns of  $\begin{bmatrix} T_{\bar{c}\bar{o}} & T_{\bar{c}o} \end{bmatrix}$  form a basis for the unobservable subspace  $\mathcal{UO}$  of the pair  $(A, C)$ ,  
and
- matrix  $T_{\bar{c}o}$  is chosen such that  $\begin{bmatrix} T_{c\bar{o}} & T_{co} & T_{\bar{c}\bar{o}} & T_{\bar{c}o} \end{bmatrix}$  is invertible.

The system given by the Kalman decomposition given by the tuple  $(\hat{A}, \hat{B}, \hat{C}, \hat{D})$  satisfies

$$\begin{aligned}\hat{A} &= T^{-1}AT \\ \hat{B} &= T^{-1}B \\ \hat{C} &= CT \\ \hat{D} &= D\end{aligned}$$

and matrices can be explicitly written as

$$\begin{aligned}\hat{A} &= \begin{bmatrix} A_{c\bar{o}} & A_{c\times} & A_{\times\bar{o}} & A_{\times\times} \\ 0 & A_{co} & 0 & A_{\times o} \\ 0 & 0 & A_{\bar{c}\bar{o}} & A_{\bar{c}\times} \\ 0 & 0 & 0 & A_{\bar{c}o} \end{bmatrix} \\ \hat{B} &= \begin{bmatrix} B_{c\bar{o}} \\ B_{co} \\ 0 \\ 0 \end{bmatrix} \\ \hat{C} &= \begin{bmatrix} 0 & C_{co} & 0 & C_{\bar{c}o} \end{bmatrix} \\ \hat{D} &= D.\end{aligned}$$

The format of the canonical decomposition leads to the conclusions:

1. the pair  $\left(\begin{bmatrix} A_{c\bar{o}} & A_{c\times} \\ 0 & A_{co} \end{bmatrix}, \begin{bmatrix} B_{c\bar{o}} \\ B_{co} \end{bmatrix}\right)$  is controllable,
2. the pair  $\left(\begin{bmatrix} A_{co} & A_{\times o} \\ 0 & A_{\bar{c}o} \end{bmatrix}, \begin{bmatrix} C_{co} \\ C_{\bar{c}o} \end{bmatrix}\right)$  is observable,
3. the triple  $(A_{co}, B_{co}, C_{co})$  is both controllable and observable, and
4. the transfer function  $C(sI - A)^{-1}B + D$  of the original system is the same as the transfer function  $C_{co}(sI - A_{co})^{-1}B_{co} + D$ .

- [ABC05] T. Alamo, J.M. Bravo, and E.F. Camacho. Guaranteed state estimation by zonotopes. *Automatica*, 41(6):1035 – 1043, 2005.
- [AH14] D. Antunes and W.P.M.H. Heemels. Rollout event-triggered control: Beyond periodic control performance. *IEEE Transactions on Automatic Control*, 59(12):3296–3311, Dec. 2014.
- [AL15] C. Altafini and G. Lini. Predictable dynamics of opinion forming for networks with antagonistic interactions. *IEEE Transactions on Automatic Control*, 60(2):342–357, Feb 2015.
- [Alt13] C. Altafini. Consensus problems on networks with antagonistic interactions. *IEEE Transactions on Automatic Control*, 58(4):935–946, April 2013.
- [Ami11] M. Amin. Guaranteeing the security of an increasingly stressed grid. *IEEE Smart Grid Newsletter*, Feb. 2011.
- [ASB07] M. Althoff, O. Stursberg, and M. Buss. Reachability analysis of linear systems with uncertain parameters and inputs. In *46th IEEE Conference on Decision and Control*, pages 726–732, Dec 2007.
- [ASP14] J. Almeida, C. Silvestre, and A.M. Pascoal. Self-triggered output feedback control of linear plants in the presence of unknown disturbances. *IEEE Transactions on Automatic Control*, 59(11):3040–3045, Nov 2014.
- [ASP15] J. Almeida, C. Silvestre, and A.M. Pascoal. Self-triggered state-feedback control of linear plants under bounded disturbances. *International Journal of Robust and Nonlinear Control*, 25(8):1230–1246, 2015.
- [ASP17] J. Almeida, C. Silvestre, and A. Pascoal. Synchronization of multiagent systems using event-triggered and self-triggered broadcasts. *IEEE Transactions on Automatic Control*, 62(9):4741–4746, Sept 2017.
- [ASS11] D. Antunes, D. Silvestre, and C. Silvestre. Average consensus and gossip algorithms in networks with stochastic asymmetric communications. In *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pages 2088–2093, Dec 2011.
- [Bar07] J. Douglas Barrett. Diagnosis and fault-tolerant control. *Technometrics*, 49(4):493–494, 2007.
- [BB97] C. Beck and P. Bendotti. Model reduction methods for unstable uncertain systems. In *36th IEEE Conference on Decision and Control*, volume 4, pages 3298–3303 vol.4, Dec 1997.
- [BB04] J. Bokor and G. Balas. Detection filter design for LPV systems – a geometric approach. *Automatica*, 40:511–518, 2004.
- [BBT<sup>+</sup>10] F. Benezit, V. Blondel, P. Thiran, J. Tsitsiklis, and M. Vetterli. Weighted gossip: Distributed averaging using non-doubly stochastic matrices. In *IEEE International Symposium on Information Theory Proceedings (ISIT)*, pages 1753–1757, June 2010.
- [BCM09] F. Bullo, J. Cortes, and S. Martinez. *Distributed Control of Robotic Networks*. Applied Mathematics Series. Princeton University Press, 2009. Electronically available at <http://coordinationbook.info>.

## Bibliography

---

- [BDX03] Stephen Boyd, Persi Diaconis, and Lin Xiao. Fastest mixing markov chain on a graph. *SIAM REVIEW*, 46:667–689, 2003.
- [Bec06] Carolyn Beck. Coprime factors reduction methods for linear parameter varying and uncertain systems. *Systems & Control Letters*, 55(3):199 – 213, 2006.
- [BGPS06] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Transactions on Information Theory*, 52(6):2508 – 2530, June 2006.
- [BHJ10] Alberto Bemporad, Maurice Heemels, and Mikael Johansson. *Networked control systems*, volume 406. Springer, 2010.
- [Bor09] E. Borel. Les probabilités dénombrables et leurs applications arithmetiques. *Rend. Circ. Mat. Palermo* (2), 27:pp. 247–271, 1909.
- [BPC<sup>+</sup>11] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multiplier. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.
- [BR71] D. Bertsekas and I. Rhodes. Recursive state estimation for a set-membership description of uncertainty. *IEEE Transactions on Automatic Control*, 16(2):117 – 128, apr 1971.
- [BRSO15] S. Bras, P. Rosa, C. Silvestre, and P. Oliveira. Fault detection and isolation in inertial measurement units based on bounding sets. *IEEE Transactions on Automatic Control*, 60(7):1933–1938, July 2015.
- [BS09] József Bokor and Zoltán Szabó. Fault detection and isolation in nonlinear systems. In *Annual Reviews in Control* 33.2, pages 113–123, 2009.
- [Can17] F. P. Cantelli. Sulla probabilità come limite della frequenza. *Atti Accad. Naz. Lincei*, 26:1:pp. 39–45, 1917.
- [CBZ10] Ruggero Carli, Francesco Bullo, and Sandro Zampieri. Quantized average consensus via dynamic coding/decoding schemes. *International Journal of Robust and Nonlinear Control*, 20(2):156–175, 2010.
- [CCS11] A. Chakraborty, J.H. Chow, and A. Salazar. A measurement-based framework for dynamic equivalencing of large power systems using wide-area phasor measurements. *IEEE Transactions on Smart Grid*, 2(1):68–81, March 2011.
- [CHPS11] V. Calderaro, C.N. Hadjicostis, A. Piccolo, and P. Siano. Failure identification in smart grids based on petri net modeling. *IEEE Transactions on Industrial Electronics*, 58(10):4613–4623, Oct 2011.
- [CHT14] Long Cheng, Zeng-Guang Hou, and Min Tan. A mean square consensus protocol for linear multi-agent systems with communication noises and fixed topologies. *IEEE Transactions on Automatic Control*, 59(1):261–267, Jan 2014.
- [CI11] K. Cai and H. Ishii. Quantized consensus and averaging on gossip digraphs. *IEEE Transactions on Automatic Control*, 56(9):2087–2100, Sept 2011.
- [CI12] Kai Cai and Hideaki Ishii. Average consensus on general strongly connected digraphs. *Automatica*, 48(11):2750 – 2761, 2012.
- [CI14] Kai Cai and H. Ishii. Average consensus on arbitrary strongly connected digraphs with time-varying topologies. *IEEE Transactions on Automatic Control*, 59(4):1066–1071, April 2014.
- [CJ14] Andrea Cristofaro and Tor Arne Johansen. Fault tolerant control allocation using unknown input observers. *Automatica*, 50(7):1891 – 1897, 2014.

- 
- [CLCD07] Mung Chiang, S.H. Low, A.R. Calderbank, and J.C. Doyle. Layering as optimization decomposition: A mathematical theory of network architectures. *Proceedings of the IEEE*, 95(1):255–312, January 2007.
- [CMB06] J. Cortes, S. Martinez, and F. Bullo. Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions. *IEEE Transactions on Automatic Control*, 51(8):1289–1298, August 2006.
- [Com05] C. Combastel. A state bounding observer for uncertain non-linear continuous-time systems based on zonotopes. In *44th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pages 7228–7234, dec. 2005.
- [CP12] Jie Chen and Ron J Patton. *Robust model-based fault diagnosis for dynamic systems*, volume 3. Springer Science & Business Media, 2012.
- [CRS15] P. Casau, P. Rosa, and C. Silvestre. FITBOX - a Fault Isolation Toolbox. *IFAC-PapersOnLine*, 48(21):283–288, 2015. 9th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes SAFEPROCESS.
- [CRT<sup>+</sup>15] P. Casau, P. Rosa, S.M. Tabatabaeipour, C. Silvestre, and J. Stoustrup. A set-valued approach to FDI and FTC of wind turbines. *IEEE Transactions on Control Systems Technology*, 23(1):245–263, Jan 2015.
- [Deg74] Morris H. Degroot. Reaching a consensus. *Journal of the American Statistical Association*, 69(345):118–121, 1974.
- [DF90] Xianchun Ding and Paul M. Frank. Fault detection via factorization approach. *Systems and Control Letters*, 14(5):431–436, 1990.
- [DGH13] A.D. Dominguez-Garcia and C.N. Hadjicostis. Distributed matrix scaling and application to average consensus in directed graphs. *IEEE Transactions on Automatic Control*, 58(3):667–681, March 2013.
- [DGHec] A.D. Dominguez-Garcia and C.N. Hadjicostis. Distributed strategies for average consensus in directed graphs. In *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pages 2124–2129, Dec.
- [DHKP97] Martin Dietzfelbinger, Torben Hagerup, Jyrki Katajainen, and Martti Penttonen. A reliable randomized algorithm for the closest-pair problem. *Journal of Algorithms*, 25(1):19–51, 1997.
- [Duc09] G. Ducard. *Fault-tolerant Flight Control and Guidance Systems: Practical Methods for Small Unmanned Aerial Vehicles*. Advances in industrial control. Springer, 2009.
- [Duc15] Guillaume Ducard. Actuator fault detection in uavs. In Kimon P. Valavanis and George J. Vachtsevanos, editors, *Handbook of Unmanned Aerial Vehicles*, pages 1071–1122. Springer Netherlands, 2015.
- [FD07] M. Farhood and G.E. Dullerud. Model reduction of nonstationary lpv systems. *IEEE Transactions on Automatic Control*, 52(2):181–196, Feb 2007.
- [FLZJ13] M. Fardad, Fu Lin, Xi Zhang, and M.R. Jovanovic. On new characterizations of social influence in social networks. In *American Control Conference (ACC)*, 2013, pages 4777–4782, June 2013.
- [FMXY12] Xi Fang, Satyajayant Misra, Guoliang Xue, and Dejun Yang. Smart grid 2014; the new and improved power grid: A survey. *IEEE Communications Surveys Tutorials*, 14(4):944–980, Fourth 2012.
- [Fri91] Noah E. Friedkin. Theoretical foundations for centrality measures. *American Journal of Sociology*, 96(6):pp. 1478–1504, 1991.
-

## Bibliography

---

- [Fri11] Noah E. Friedkin. A formal theory of reflected appraisals in the evolution of power. *Administrative Science Quarterly*, 56(4):501–529, 2011.
- [Fri15] N. E. Friedkin. The problem of social control and coordination of complex systems in sociology: A look at the community cleavage problem. *IEEE Control Systems*, 35(3):40–51, June 2015.
- [FRTI13] Paolo Frasca, Chiara Ravazzi, Roberto Tempo, and Hideaki Ishii. Gossips and prejudices: Ergodic randomized dynamics in social networks. *IFAC Proceedings Volumes*, 46(27):212 – 219, 2013.
- [FZ08] F. Fagnani and S. Zampieri. Randomized consensus algorithms over large scale networks. *IEEE Journal on Selected Areas in Communications*, 26(4):634–649, May 2008.
- [FZ09] Fabio Fagnani and Sandro Zampieri. Average consensus with packet drop communication. *SIAM Journal on Control and Optimization*, 48(1):102–133, 2009.
- [GBG<sup>+</sup>11] A. Giani, E. Bitar, M. Garcia, M. McQueen, P. Khargonekar, and K. Poolla. Smart grid data integrity attacks: characterizations and countermeasures. In *IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 232–237, Oct 2011.
- [GG76] M.S. Grewal and K. Glover. Identifiability of linear and nonlinear dynamical systems. *IEEE Transactions on Automatic Control*, 21(6):833–837, 1976.
- [GYH17] Xiaohua Ge, Fuwen Yang, and Qing-Long Han. Distributed networked control systems: A brief overview. *Information Sciences*, 380:117 – 131, 2017.
- [HC14] C.N. Hadjicostis and T. Charalambous. Average consensus in the presence of delays in directed graph topologies. *IEEE Transactions on Automatic Control*, 59(3):763–768, March 2014.
- [HJT12] WPMH Heemels, Karl Henrik Johansson, and Paulo Tabuada. An introduction to event-triggered and self-triggered control. In *51st IEEE Conference on Decision and Control, Maui, HI, USA.*, pages 3270–3285, 2012.
- [HK02] Rainer Hegselmann and Ulrich Krause. Opinion dynamics and bounded confidence models, analysis and simulation. *Journal of Artificial Societies and Social Simulation*, 5(3):2, 2002.
- [HKKS10] Inseok Hwang, Sungwan Kim, Youdan Kim, and C.E. Seah. A survey of fault detection, isolation, and reconfiguration methods. *IEEE Transactions on Control Systems Technology*, 18(3):636 –653, may 2010.
- [HKY98] H. Hammouri, M. Kinnaert, and E.H. El Yaagoubi. Fault detection and isolation for state affine systems. *European Journal of Control*, 4(1):2 – 16, 1998.
- [HNX07] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu. A Survey of Recent Results in Networked Control Systems. *Proceedings of the IEEE*, 95(1):138–162, January 2007.
- [Hos82] G.H. Hostetter. Ongoing deadbeat observers for linear time-varying systems. In *American Control Conference, 1982*, pages 1099–1101, June 1982.
- [HSB08] W. P. M. H. Heemels, J. H. Sandee, and P. P. J. Van Den Bosch. Analysis of event-driven controllers for linear systems. *International Journal of Control*, 81(4):571–590, 2008.
- [HSJ14] J.M. Hendrickx, G. Shi, and K.H. Johansson. Finite-time consensus using stochastic matrices with positive diagonals. *IEEE Transactions on Automatic Control*, PP(99):1–1, 2014.



- 
- [IK90] Y. E. Ioannidis and Younkyung Kang. Randomized algorithms for optimizing large join queries. *SIGMOD Rec.*, 19(2):312–321, May 1990.
- [IT10] H. Ishii and R. Tempo. Distributed randomized algorithms for the pagerank computation. *IEEE Transactions on Automatic Control*, 55(9):1987–2002, Sept 2010.
- [JKJJ08] B. Johansson, T. Keviczky, M. Johansson, and K.H. Johansson. Subgradient methods and consensus algorithms for solving convex optimization problems. In *47th IEEE Conference on Decision and Control*, pages 4185–4190, December 2008.
- [JLM03] A. Jadbabaie, Jie Lin, and A. S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, June 2003.
- [JMFB13] Peng Jia, Anahita Mirtabatabaei, Noah E. Friedkin, and Francesco Bullo. On the dynamics of influence networks via reflected appraisal. In *American Control Conference (ACC), 2013*, pages 1249–1254, 2013.
- [KDG03] D. Kempe, A. Dobra, and J. Gehrke. Gossip-based computation of aggregate information. In *44th IEEE Symposium on Foundations of Computer Science*, pages 482–491, Oct. 2003.
- [Kez11] M. Kezunovic. Smart fault location for smart grids. *IEEE Transactions on Smart Grid*, 2(1):11–22, March 2011.
- [KG87] S. Keerthi and E. Gilbert. Computation of minimum-time feedback control laws for discrete-time systems with state-control constraints. *IEEE Transactions on Automatic Control*, 32(5):432 – 435, may 1987.
- [KKPD13] Z. Kan, J. Klotz, E.L. Pasiliao, and W.E. Dixon. Containment control for a directed social network with state-dependent connectivity. In *American Control Conference (ACC), 2013*, pages 1950–1955, 2013.
- [KMMS97] Kim Potter Kihlstrom, L. E. Moser, and P. M. Melliar-Smith. Solving consensus in a byzantine environment using an unreliable fault detector. In *Proceedings of the International Conference on Principles of Distributed Systems (OPODIS)*, pages 61–75, 1997.
- [Kra97] David Krackhardt. Organizational viscosity and the diffusion of controversial innovations. *The Journal of Mathematical Sociology*, 22(2):177–199, 1997.
- [KV06] A. A. Kurzhanskiy and P. Varaiya. Ellipsoidal toolbox. Technical Report UCB/EECS-2006-46, EECS Department, University of California, Berkeley, May 2006.
- [Lev96] William S Levine. *The control handbook*. CRC press, 1996.
- [Lew96] A. S. Lewis. Convex analysis on the hermitian matrices. *SIAM Journal on Optimization*, 6:164–177, 1996.
- [LM12] Ji Liu and A.S. Morse. Asynchronous distributed averaging using double linear iterations. In *American Control Conference (ACC), 2012*, pages 6620–6625, June 2012.
- [LWZ14] Tao Li, Fuke Wu, and Ji-Feng Zhang. Multi-agent consensus with relative-state-dependent measurement noises. *IEEE Transactions on Automatic Control*, 59(9):2463–2468, Sept 2014.
- [MCHL14] K. Manandhar, Xiaojun Cao, Fei Hu, and Yao Liu. Detection of faults and attacks including false data injection attack in smart grid using kalman filter. *IEEE Transactions on Control of Network Systems*, 1(4):370–379, Dec 2014.
-

## Bibliography

---

- [ME10] A.R. Metke and R.L. Ekl. Security technology for smart grid networks. *IEEE Transactions on Smart Grid*, 1(1):99–107, June 2010.
- [ME14] P.P. Menon and C. Edwards. Robust fault estimation using relative information in linear multi-agent networks. *IEEE Transactions on Automatic Control*, 59(2):477–482, Feb 2014.
- [MGB05] Andres Marcos, Subhabrata Ganguli, and Gary J. Balas. An application of  $H_\infty$  fault detection and isolation to a transport aircraft. *Control Engineering Practice*, 13(1):105 – 119, 2005.
- [Moo66] Ramon E Moore. *Interval analysis*. Prentice-Hall series in automatic computation. Prentice-Hall, Englewood Cliffs, NJ, 1966.
- [Mor04] L. Moreau. Stability of continuous-time distributed consensus algorithms. In *43rd IEEE Conference on Decision and Control*, volume 4, pages 3998 – 4003 Vol.4, dec. 2004.
- [MR10] Rajeev Motwani and Prabhakar Raghavan. Randomized algorithms. In Mikhail J. Atallah and Marina Blanton, editors, *Algorithms and Theory of Computation Handbook*, pages 12–12. Chapman & Hall/CRC, 2010.
- [MT08] M. Mazo and P. Tabuada. On event-triggered and self-triggered control over sensor/actuator networks. In *47th IEEE Conference on Decision and Control*, pages 435–440, Dec 2008.
- [MT11] M. Mazo and P. Tabuada. Decentralized event-triggered control over wireless sensor/actuator networks. *IEEE Transactions on Automatic Control*, 56(10):2456–2461, Oct 2011.
- [Mul94] K. Mulmuley. *Computational Geometry: An Introduction through Randomized Algorithms*. Prentice-Hall, NJ, 1994.
- [MV91] M. Milanese and A. Vicino. Optimal estimation theory for dynamic systems with set membership uncertainty: An overview. *Automatica*, 27(6):997 – 1009, 1991.
- [MV09] P. Massioni and M. Verhaegen. Distributed control for identical dynamically coupled systems: A decomposition approach. *IEEE Transactions on Automatic Control*, 54(1):124–135, Jan 2009.
- [NVR08] S. Narasimhan, P. Vachhani, and R. Rengaswamy. New nonlinear residual feedback observer for fault diagnosis in nonlinear systems. *Automatica*, 44:2222–2229, 2008.
- [OSM04] R. Olfati-Saber and R.M. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49(9):1520 – 1533, sept. 2004.
- [oW15] University of Washigton, March 2015.
- [Pat97] Ron J. Patton. Fault-tolerant control systems: The 1997 situation. In *IFAC symposium on fault detection supervision and safety for technical processes*, volume 3, 1997.
- [PBB11] F. Pasqualetti, A. Bicchi, and F. Bullo. A graph-theoretical characterization of power network vulnerabilities. In *American Control Conference (ACC)*, 2011, pages 3918–3923, June 2011.
- [PBB12] F. Pasqualetti, A. Bicchi, and F. Bullo. Consensus computation in unreliable networks: A system theoretic approach. *IEEE Transactions on Automatic Control*, 57(1):90 –104, jan. 2012.

- 
- [PBEA10] S. Patterson, B. Bamieh, and A. El Abbadi. Convergence rates of distributed average consensus with stochastic link failures. *IEEE Transactions on Automatic Control*, 55(4):880–892, April 2010.
  - [PDB11] Fabio Pasqualetti, Florian Dorfler, and F. Bullo. Cyber-physical attacks in power networks: Models, fundamental limitations and monitor design. In *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pages 2195–2201, Dec 2011.
  - [Pol02] David Pollard. *A user's guide to measure theoretic probability*, volume 8. Cambridge University Press, 2002.
  - [PS07] M. Porfiri and D.J. Stilwell. Consensus seeking over random weighted directed graphs. *IEEE Transactions on Automatic Control*, 52(9):1767–1773, sept. 2007.
  - [PT17] Anton V. Proskurnikov and Roberto Tempo. A tutorial on modeling and analysis of dynamic social networks. part i. *Annual Reviews in Control*, 43:65–79, 2017.
  - [REZ12] T. Raissi, D. Efimov, and A. Zolghadri. Interval state estimation for a class of nonlinear systems. *IEEE Transactions on Automatic Control*, 57(1):260–265, Jan 2012.
  - [RFTI15] C. Ravazzi, P. Frasca, R. Tempo, and H. Ishii. Ergodic randomized algorithms and dynamics over networks. *IEEE Transactions on Control of Network Systems*, 2(1):78–87, March 2015.
  - [RGTC01] Sridharan Ranganathan, AlanD. George, RobertW. Todd, and MatthewC. Chidester. Gossip-style failure detection and distributed consensus for scalable heterogeneous clusters. *Cluster Computing*, 4(3):197–209, 2001.
  - [RMH98] Robbert Renesse, Yaron Minsky, and Mark Hayden. A gossip-style failure detection service. In Nigel Davies, Seitz Jochen, and Kerry Raymond, editors, *Middleware'98*, pages 55–70. Springer London, 1998.
  - [RNEV08] R. Rajagopal, XuanLong Nguyen, S.C. Ergen, and P. Varaiya. Distributed online simultaneous fault detection for multiple sensors. In *International Conference on Information Processing in Sensor Networks (IPSN)*, pages 133–144, April 2008.
  - [Ros11] Paulo Rosa. *Multiple-Model Adaptive Control of Uncertain LPV Systems*. PhD thesis, Technical University of Lisbon, Lisbon, Portugal, 2011.
  - [RPK92] R. Ravi, A.M. Pascoal, and P.P. Khargonekar. Normalized coprime factorizations for linear time-varying systems. *Systems & Control Letters*, 18(6):455–465, 1992.
  - [RS00] Wilson J. Rugh and Jeff S. Shamma. Research on gain scheduling. *Automatica*, 36(10):1401–1425, 2000.
  - [RS11] P. Rosa and C. Silvestre. On the distinguishability of discrete linear time-invariant dynamic systems. In *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pages 3356–3361, Dec 2011.
  - [RS13] Paulo Rosa and Carlos Silvestre. Fault detection and isolation of LPV systems using set-valued observers: An application to a fixed-wing aircraft. *Control Engineering Practice*, 21(3):242–252, 2013.
  - [RS14] Paulo Rosa and Carlos Silvestre. Multiple-model adaptive control using set-valued observers. *International Journal of Robust and Nonlinear Control*, 24(16):2490–2511, 2014.
  - [RSA14] Paulo Rosa, Carlos Silvestre, and Michael Athans. Model falsification using set-valued observers for a class of discrete-time dynamic systems: a coprime factorization approach. *International Journal of Robust and Nonlinear Control*, 24(17):2928–2942, 2014.
-

## Bibliography

---

- [RSSA10] P. Rosa, C.J. Silvestre, J.S. Shamma, and M. Athans. Fault detection and isolation of LTV systems using set-valued observers. *49th IEEE Conference on Decision and Control*, Atlanta, Georgia, USA., pages 768–773, December 2010.
- [Sau05] Dominique Sauter. Diagnosis and fault-tolerant control m. blanke, m. kinnaert, j. lunze and m. staroswiecki, springer-verlag: Berlin, 2003, 571 pp, isbn 3-540-01056-4. *International Journal of Robust and Nonlinear Control*, 15(3):151–154, 2005.
- [SC16] J. Su and W. H. Chen. Fault diagnosis for vehicle lateral dynamics with robust threshold. In *2016 IEEE International Conference on Industrial Technology (ICIT)*, pages 1777–1782, March 2016.
- [Sch68] F. Schweppe. Recursive state estimation: Unknown but bounded errors and system inputs. *IEEE Transactions on Automatic Control*, 13(1):22 – 28, feb 1968.
- [Sch73] F. Schweppe. *Uncertain Dynamic Systems*. Prentice-Hall, 1973.
- [Sch04] Ernst Scholtz. *Observer-based monitors and distributed wave controllers for electromechanical disturbances in power systems*. PhD thesis, Massachusetts Institute of Technology, 2004.
- [SH11] S. Sundaram and C.N. Hadjicostis. Distributed function calculation via linear iterative strategies in the presence of malicious agents. *IEEE Transactions on Automatic Control*, 56(7):1495–1508, July 2011.
- [SHGE14] T. Sadikhov, W.M. Haddad, R. Goebel, and M. Egerstedt. Set-valued protocols for almost consensus of multiagent systems with uncertain interagent communication. In *American Control Conference (ACC)*, 2014, pages 4002–4007, June 2014.
- [SJ13] Guodong Shi and Karl Henrik Johansson. Convergence of distributed averaging and maximizing algorithms part ii: State-dependent graphs. In *American Control Conference (ACC)*, 2013, pages 6875–6880, 2013.
- [SP98] Peter W Sauer and MA Pai. *Power system dynamics and stability*, volume 4. Prentice Hall Upper Saddle River, NJ, 1998.
- [SRC<sup>+</sup>13] D. Silvestre, P. Rosa, R. Cunha, J.P. Hespanha, and C. Silvestre. Gossip average consensus in a byzantine environment using stochastic set-valued observers. In *52nd IEEE Conference on Decision and Control*, pages 4373–4378, Dec 2013.
- [SRHS14] D. Silvestre, P. Rosa, J.P. Hespanha, and C. Silvestre. Finite-time average consensus in a byzantine environment using set-valued observers. In *American Control Conference (ACC)*, 2014, pages 3023–3028, June 2014.
- [SRHS15a] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre. Distributed fault detection using relative information in linear multi-agent networks. *IFAC-PapersOnLine*, 48(21):446–451, 2015. 9th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes SAFEPROCESS 20, Paris, 2-4 September 2015.
- [SRHS15b] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre. Finite-time convergence policies in state-dependent social networks. In *American Control Conference (ACC)*, 2015, Chicago, Illinois, USA., July 2015.
- [SRHS15c] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre. Self-triggered set-valued observers. In *European Control Conference (ECC)*, pages 3647–3652, July 2015.
- [SRHS15d] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre. Set-consensus using set-valued observers. In *American Control Conference (ACC)*, 2015, Chicago, Illinois, USA., July 2015.

- 
- [SRHS17a] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre. Fault detection for LPV systems using set-valued observers: A coprime factorization approach. *Systems & Control Letters*, 106:32 – 39, 2017.
  - [SRHS17b] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre. Set-based fault detection and isolation for detectable linear parameter-varying systems. *International Journal of Robust and Nonlinear Control*, 27(18):4381–4397, 2017. rnc.3814.
  - [SRHS17c] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre. Stochastic and deterministic fault detection for randomized gossip algorithms. *Automatica*, 78:46 – 60, 2017.
  - [SRHS18] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre. Self-triggered and event-triggered set-valued observers. *Information Sciences*, 426:61 – 86, 2018.
  - [SRHSe] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre. Stochastic and deterministic state-dependent social networks. *IEEE Transactions on Automatic Control*, Conditionally Accepted.
  - [SRHSe] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre. Broadcast and gossip stochastic average consensus algorithms in directed topologies. *IEEE Transactions on Control of Network Systems*, In review.
  - [SRMB16] Joseph K. Scott, Davide M. Raimondo, Giuseppe Roberto Marseglia, and Richard D. Braatz. Constrained zonotopes: A new tool for set-based estimation and fault detection. *Automatica*, 69:126 – 136, 2016.
  - [SST93] Eldar Shafir, Itamar Simonson, and Amos Tversky. Reason-based choice. *Cognition*, 49(1–2):11 – 36, 1993.
  - [ST99] J.S. Shamma and Kuang-Yang Tu. Set-valued observers and optimal disturbance rejection. *IEEE Transactions on Automatic Control*, 44(2):253 – 264, feb 1999.
  - [TBA86] J. Tsitsiklis, D. Bertsekas, and M. Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE Transactions on Automatic Control*, 31(9):803 – 812, September 1986.
  - [TC03] Yodyium Tipsuwan and Mo-Yuen Chow. Control methodologies in networked control systems. *Control Engineering Practice*, 11(10):1099 – 1111, 2003. Special Section on Control Methods for Telecommunication.
  - [Tel82] J. Telgen. Minimal representation of convex polyhedral sets. *Journal of Optimization Theory and Applications*, 38(1):1–24, 1982.
  - [TFNM13] B. Touri, F. Fardnoud, A. Nedic, and O. Milenkovic. A general framework for distributed vote aggregation. In *American Control Conference*, pages 3827–3832, June 2013.
  - [TKA<sup>+</sup>12] Giang Tran, A. Kiani, A. Annaswamy, Y. Sharon, A.L. Motto, and A. Chakraborty. Necessary and sufficient conditions for observability in power systems. In *IEEE Innovative Smart Grid Technologies (ISGT)*, pages 1–8, Jan 2012.
  - [TN14] B. Touri and A. Nedic. Product of random stochastic matrices. *IEEE Transactions on Automatic Control*, 59(2):437–448, Feb 2014.
  - [TSJ08] A. Tahbaz-Salehi and A. Jadbabaie. A necessary and sufficient condition for consensus over random networks. *IEEE Transactions on Automatic Control*, 53(3):791–795, April 2008.
  - [UR12] D. Ustebay and M. Rabbat. Efficiently reaching consensus on the largest entries of a vector. In *51st IEEE Conference on Decision and Control*, pages 56–61, 2012.
-

## Bibliography

---

- [WDAN02] Gérard Weisbuch, Guillaume Deffuant, Frédéric Amblard, and Jean-Pierre Nadal. Meet, discuss, and segregate! *Complexity*, 7(3):55–63, 2002.
- [Wil76] Alan S. Willsky. A survey of design methods for failure detection in dynamic systems. *Automatica*, 12(6):601 – 611, 1976.
- [Wil91] David Williams. *Probability with martingales*. Cambridge University Press, Cambridge, 1991.
- [Wit68] H. Witsenhausen. Sets of possible states of linear systems given perturbed observations. *IEEE Transactions on Automatic Control*, 13(5):556 – 558, oct 1968.
- [WQF15] Yanling Wei, Jianbin Qiu, and Shasha Fu. Mode-dependent nonrational output feedback control for continuous-time semi-markovian jump systems with time-varying delay. *Nonlinear Analysis: Hybrid Systems*, 16:52 – 71, 2015.
- [WQKW14] Yanling Wei, Jianbin Qiu, Hamid Reza Karimi, and Mao Wang. Filtering design for two-dimensional markovian jump systems with state-delays and deficient mode information. *Information Sciences*, 269:316 – 331, 2014.
- [WYB02] G. C. Walsh, Hong Ye, and L. G. Bushnell. Stability analysis of networked control systems. *IEEE Transactions on Control Systems Technology*, 10(3):438–446, May 2002.
- [ZDG96] Kemin Zhou, John C. Doyle, and Keith Glover. *Robust and Optimal Control*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.
- [ZHY16] X. M. Zhang, Q. L. Han, and X. Yu. Survey on recent advances in networked control systems. *IEEE Transactions on Industrial Informatics*, 12(5):1740–1752, Oct 2016.
- [ZJ14] Ze Zhang and Imad M. Jaimoukha. On-line fault detection and isolation for linear discrete-time uncertain systems. *Automatica*, 50(2):513 – 518, 2014.
- [ZMXZ15] X. Zhao, C. Ma, X. Xing, and X. Zheng. A stochastic sampling consensus protocol of networked euler-lagrange systems with application to two-link manipulator. *IEEE Transactions on Industrial Informatics*, 11(4):907–914, Aug 2015.