Stability Analysis of Networked Control Systems: Comparison & Implementation

J.J.C. van Schendel

DCT 2009.099

Master’s thesis

Committee:  prof. dr. ir. M. Steinbuch (Chairman)
            ir. M.C.F. Donkers (Coach)
            dr. ir. W.P.M.H. Heemels (Coach)
            dr. ir. N. van de Wouw (Coach)
            dr. M. Lazar
            N.W. Bauer, MSc

Eindhoven University of Technology
Department of Mechanical Engineering
Control Systems Technology Group

Eindhoven, November 5, 2009
Abstract

Networked control systems (NCSs) are receiving more and more attention these days. NCSs are systems where the control loop, in general consisting of a continuous-time plant and a discrete-time controller, is closed over a communication network. The interest for NCSs is motivated by the many benefits they offer such as ease of maintenance and installation, the large flexibility due to the use of distributed elements and the low costs. However, also some disadvantages arise. These disadvantages are caused by the unreliability and shared use of the network, resulting in time-varying delays, packet dropouts and variations in the sampling interval. There is an extensive literature available on the modelling of NCSs, including these network phenomena. However, most of the available work focusses only on one of these phenomena, while ignoring the others. Clearly, it is important to consider the combined presence of dropouts, time-varying sampling intervals and time-varying delays, as in any practical NCS these typically occur simultaneously.

In this thesis, an overview is presented of the literature that considers all three network effects using a discrete-time modelling approach. This is done by performing a thorough comparison in terms of modelling accuracy, conservatism and complexity of the stability analysis methods for NCSs. In particular, three methods for modelling dropouts are considered, namely (i) modelling dropouts as prolongation of the sampling interval, (ii) modelling dropouts as prolongation of the delay and (iii) a new dropout modelling approach is proposed that is based on modelling dropouts explicitly using automata. In this work, extensions of the existing methods are given that include the considerations of varying sampling intervals and the usage of parameter-dependent Lyapunov functions to perform the stability analysis. Based on polytopic overapproximations of the resulting discrete-time NCS models, LMI-based stability conditions are provided. Herewith, the extensions of the existing approaches and the newly proposed method are compared in terms of modelling accuracy, conservatism and complexity of the stability analysis.

The research presented in this thesis, also includes the design of a Matlab toolbox, in which two modelling approaches, i.e. method (ii) and (iii), and two ways to overapproximate the resulting discrete-time NCS models, i.e. based on the real Jordan form and the Cayley-Hamilton theorem, are implemented. This Matlab toolbox is useful for stability analysis of NCSs and can be used to perform further research.
# Contents

Abstract i

1 Introduction 1
  1.1 Contributions of this Thesis .......................... 1
  1.2 Outline of this Thesis .................................. 2

2 Modelling and Analysis of NCSs 3
  2.1 Description of the NCS .................................. 3
     2.1.1 Real Jordan Form Approach ......................... 4
     2.1.2 Cayley-Hamilton Approach .......................... 8
  2.2 Stability Analysis Using Lyapunov Functions ................. 11
  2.3 Controller Synthesis ................................ 12

3 Comparison of Dropout Modelling Approaches 13
  3.1 Introduction ........................................... 14
  3.2 The NCS Model .......................................... 15
     3.2.1 Dropouts as Prolongation of Delays ..................... 17
     3.2.2 Dropouts as Prolongation of Sampling Intervals .......... 18
     3.2.3 Explicit Dropout Modelling .......................... 19
  3.3 Overapproximation of Model .............................. 19
  3.4 Stability Analysis of NCS ................................ 21
  3.5 Discussion ............................................ 21
  3.6 Illustrative Example .................................... 23
  3.7 Conclusions ........................................... 25

4 The Matlab Toolbox 27
  4.1 Matlab Implementation Without Dropouts ...................... 30
     4.1.1 Overapproximation in the Matlab Toolbox ................. 30
     4.1.2 Overapproximation Using the Real Jordan Form .......... 31
     4.1.3 Overapproximation Using the Cayley-Hamilton Theorem .... 32
     4.1.4 Apply Available Controller to the Overapproximated Model ... 33
     4.1.5 Stability Analysis in the Matlab Toolbox ............... 33
     4.1.6 Controller Synthesis ................................ 33
  4.2 Matlab Implementation With Dropouts ...................... 34
  4.3 Using the Matlab Toolbox ................................ 34

5 Numerical Comparison 37
  5.1 Comparing the Overapproximations ......................... 38
  5.2 Results of the Comparison ................................ 38

6 Conclusions and Future Work 41
  6.1 Conclusions ........................................... 41
  6.2 Recommendations for Future Research ....................... 42
Chapter 1

Introduction

The complexity of mechatronic systems, such as wafersteppers, electron microscopes and copiers, increased rapidly over the past decades. A major reason for this increase in complexity is the fact that functionality and performance requirements are ever increasing. The cost price however should be kept as low as possible to ensure a competitive position in the market. During the design process, many choices have to be made that influence the final product. A way to make well-considered decisions during hightech systems design, is to use models that predict the system behaviour and conditions that indicate how and when these models can be used. Typically, to obtain the most optimal design and to obtain the models that predict the total system behaviour, different disciplines have to work together closely. The research presented in this thesis focusses on the relation between two of these disciplines, namely control engineering and software engineering. In particular, we focus on the information flow between the system to be controlled and the processor, on which the controller is implemented, which is of interest for both control engineering as well as software engineering. This coupling is described in the literature of so-called networked control systems (NCSs).

NCSs are systems where the control loop, in general consisting of a continuous-time plant and a discrete-time controller, is closed over a communication network. Over the past decade, the interest in NCSs has increased rapidly [1, 25, 33]. The advantages that an NCS offers are its flexible architecture, due to the use of distributed elements, and a reduction of installation and maintenance costs. NCSs can easily be modified or upgraded by adding sensors, actuators and controllers with relatively low cost and without making any major changes in their overall structure. Along with the advantages that NCSs induce, several challenges emerged. These challenges are caused by the unreliability and shared use of the network, resulting in time-varying delays, packet dropouts and variations in the sampling interval. All these phenomena make the overall behaviour of an NCS complex and NCSs have to meet the requirements on system stability, performance and reliability in spite of the presence of these uncertainties. So far, most literature discusses the combined presence of a selection of these phenomena, whereas this thesis considers NCSs that include all three network phenomena simultaneously.

1.1 Contributions of this Thesis

Over the past decade, an abundance of literature has become available on the modelling of NCSs, in which the effects of time-varying delays, time-varying sampling intervals and packet dropouts are studied, using both discrete-time and continuous-time modelling frameworks [28, 15, 31, 22, 4, 10]. Within any of these frameworks, the third phenomenon (packet dropouts) is modelled in many different ways. To get overview of the methods to model dropouts and to point out the advantages and disadvantages of each method, we focus on two existing modelling methods for the discrete-time framework. Moreover, a new discrete-time modelling method, based on hybrid automata
is proposed and compared with the other two techniques in terms of modelling accuracy, conservatism and complexity of the stability analysis. However, before a thorough comparison can be made on the level of stability analysis, some obstacles have to be taken. One of these obstacles is the so-called overapproximation of the uncertain discrete-time NCS model by polytopic discrete-time models. The need for this overapproximation arises as there are an infinite number of sequences of delays and sampling intervals. Furthermore, the uncertain parameters appear nonlinearly. It is therefore impossible to perform stability analysis directly on the uncertain discrete-time NCS models and the models need to be overapproximated such that they are amendable for stability analysis.

To perform a comparison of the different dropout modelling approaches and to make the theory of NCSs analysis more accessible for both industrial and academic researchers, a Matlab toolbox is developed that includes all the steps to come to a complete NCS model and that performs the subsequent stability analysis and controller design. Such a toolbox is not yet available and would be useful for any (non-)expert that wants to analyse NCSs. The Matlab toolbox designed in this thesis includes two different ways to model NCSs that incorporate dropouts, time-varying delays and time-varying sampling intervals. Moreover, all the steps that need to be taken to perform a complete NCS analysis are implemented in the toolbox, so that any plant model can be analysed.

1.2 Outline of this Thesis

This thesis consists of six chapters. In Chapter 2 an NCS model that incorporates time-varying delays and time-varying sampling intervals is discussed. This serves the purpose of providing preliminary knowledge and for understanding the difficulties in designing and analysing such an NCS model. Chapter 3 studies three different dropout modelling methods and compares them in terms of modelling accuracy, conservatism and complexity of the stability analysis. Chapter 4 discusses the designed Matlab toolbox by going through all the steps that are implemented in Matlab. In Chapter 5, three examples are used to perform a numerical comparison of the complete analysis of an NCS using the Matlab toolbox. Finally, some concluding remarks are given and some possible future research topics are presented in Chapter 6.
Chapter 2

Modelling and Analysis of NCSs

As explained in the introduction, the use of networks induces communication imperfections and constraints. This thesis focusses on three phenomena, namely varying sampling intervals, varying transmission delays and packet dropouts caused by the unreliability of the network. In this chapter, the design of an NCS model that incorporates the first two network phenomena, i.e., varying sampling intervals and delays, is discussed. It is important to understand how these phenomena influence the closed-loop stability and performance properties since they are present in any practical communication network. As a consequence, there is a need for analysis and synthesis methods that explicitly include these network-induced phenomena.

2.1 Description of the NCS

The NCS as considered in this chapter, includes delays that are smaller than the uncertain and time-varying sampling intervals. The NCS is depicted schematically in Fig. 2.1 and consists of a linear continuous-time plant

\[ \dot{x}(t) = Ax(t) + Bu(t), \]  

with \( A \in \mathbb{R}^{n \times n} \) and \( B \in \mathbb{R}^{n \times m} \), and a discrete-time controller, which are connected over a communication network that induces network delays (\( \tau^{sc} \) and \( \tau^{ca} \)). The state measurements are sampled at the sampling time instants \( s_k = \sum_{i=0}^{k-1} h_i \) for all \( k \geq 1 \) and \( s_0 = 0 \), where \( h_k > 0 \) is the time-varying sampling interval. We denote by \( x_k := x(s_k) \) the \( k \)th sampled value of the state \( x \) and by \( u_k = u(s_k) \) the control value at sample instants \( s_k \).

In the model, both the varying computation time (\( \tau^c \)), needed to compute the \( k \)th control value, and the network-induced delays, i.e. the sensor-to-controller delay (\( \tau^{sc} \)) and the controller-to-actuator delay (\( \tau^{ca} \)), are taken into account. In this thesis, it is assumed that the sensor acts in a time-driven fashion (i.e. sampling occurs at the times \( s_k \)) and that both the controller and the actuator act in an event-driven fashion (i.e. responding instantaneously to newly arrived data). Under these assumptions, all three delays can be captured by a single delay \( \tau_k := \tau^{sc}_k + \tau^c_k + \tau^{ca}_k \), \( k \in \mathbb{N} \). Furthermore, it is also assumed that \( \tau_k \in [\tau_{\text{min}}, \tau_{\text{max}}] \), with \( 0 \leq \tau_{\text{min}} \leq \tau_{\text{max}} \leq h_k \), \( h_k \in [h_{\text{min}}, h_{\text{max}}] \), with \( 0 < h_{\text{min}} \leq h_{\text{max}} \), and that \( \tau_k \leq h_k \) (there are also models available that consider large delays) for all \( k \in \mathbb{N} \), i.e., each \((h, \tau) \in \Theta \), in which

\[ \Theta = \{(h, \tau) \in \mathbb{R}^2 \mid h \in [h_{\text{min}}, h_{\text{max}}], \tau \in [\tau_{\text{min}}, \min\{h, \tau_{\text{max}}\}] \}. \]  

Finally, the zero-order-hold (ZOH) function in Fig. 2.1 transforms the discrete-time control input \( u_k \) to a continuous-time control input \( u^*(t) \). To be precise, \( u^*(t) = u_k \) when \( t \in [s_k + \tau_k, s_{k+1} + \tau_{k+1}) \) for all \( k \in \mathbb{N} \). If the linear plant from (2.1) is now discretised at the sampling times \( s_k \), \( k \in \mathbb{N} \), we obtain

\[ x_{k+1} = e^{Ah_k} x_k + \int_{0}^{h_k - \tau_k} e^{As} B u_k + \int_{h_k - \tau_k}^{h_k} e^{As} B u_{k-1}. \]  

(2.3)
Performing stability analysis directly on the uncertain system (2.4) is difficult, because the uncertainties \( \tau \) and \( h \) appear in exponential terms and integrals thereof. To render the formulation in (2.4) amendable for stability analysis, overapproximation techniques are employed in the literature to embed the original model (as tight as possible) in a larger model that has nice structural properties. The resulting overapproximated models are suitable for the application of stability analysis methods.

In the literature, many different approaches to overapproximate an NCS exist, in which polytopic embeddings of the uncertain system are proposed, such as overapproximation based on gridding (see, e.g., [8, 9, 23]), on the Taylor series expansion (see [12]), on the real Jordan form (see [22, 4, 21]) and on the Cayley-Hamilton theorem (see [11]). In this thesis, the focus is on the latter two approaches (overapproximation based on the real Jordan form and the Cayley-Hamilton theorem) and the aim is to present a thorough comparison of these two different techniques on the basis of the approximation error and conservatism introduced by these techniques.

### 2.1.1 Real Jordan Form Approach

In this section the real Jordan form is used to overapproximate the discrete-time NCS model (2.4) into a polytopic model which has nice structural properties. The advantage of using the real Jordan form is that an explicit expression can be derived for the matrix exponential and integral into a polytopic model which has nice structural properties. The resulting overapproximated models are suitable for the application of stability analysis methods.

Now using the lifted state vector \( \xi_k = \begin{bmatrix} x_k^T & u_{k-1}^T \end{bmatrix}^T \), that includes the current system state and past system input, the following model describing the NCS is obtained

\[
\xi_{k+1} = \begin{bmatrix} e^{Ah_k} & \int_{h_k}^{h_{k+1}} e^{As}dsB \\ 0 & 0 \end{bmatrix} \xi_k + \begin{bmatrix} \int_{h_k}^{h_{k+1}} e^{As}dsB \\ 0 \end{bmatrix} u_k.
\] (2.4)

Figure 2.1: Schematic overview of the NCS

Performing stability analysis directly on the uncertain system (2.4) is difficult, because the uncertainties \( \tau \) and \( h \) appear in exponential terms and integrals thereof. To render the formulation in (2.4) amendable for stability analysis, overapproximation techniques are employed in the literature to embed the original model (as tight as possible) in a larger model that has nice structural properties. The resulting overapproximated models are suitable for the application of stability analysis methods.

In the literature, many different approaches to overapproximate an NCS exist, in which polytopic embeddings of the uncertain system are proposed, such as overapproximation based on gridding (see, e.g., [8, 9, 23]), on the Taylor series expansion (see [12]), on the real Jordan form (see [22, 4, 21]) and on the Cayley-Hamilton theorem (see [11]). In this thesis, the focus is on the latter two approaches (overapproximation based on the real Jordan form and the Cayley-Hamilton theorem) and the aim is to present a thorough comparison of these two different techniques on the basis of the approximation error and conservatism introduced by these techniques.

\[
\xi_{k+1} = \left( F_0 + \sum_{i=1}^{\varsigma} \alpha_i(h_k, \tau_k)F_i \right) \xi_k + \left( G_0 + \sum_{i=1}^{\varsigma} \alpha_i(h_k, \tau_k)G_i \right) u_k,
\] (2.5)

where \( \varsigma \) is the number of time-varying functions \( \alpha_i \) given by \( 2\nu \), due to the two varying parameters \( h \) and \( \tau \), with \( \nu \leq n \), where \( n \) is the dimension of the state vector \( x \). We have \( \nu = n \) when each distinct eigenvalue of \( A \) corresponds to one Jordan block only and \( \nu < n \) otherwise. The functions \( \alpha_i \) are typically of the form \( (h_k - \tau_k)^{j-1}e^{\lambda(h_k - \tau_k)} \) if \( \lambda \) is a real and nonzero eigenvalue of \( A \), \( (h_k - \tau_k)^j \) if \( \lambda \) is zero, and \( (h_k - \tau_k)^{j-1}e^{(h_k - \tau_k)\sin(b(h_k - \tau_k))} \) or \( (h_k - \tau_k)^{j-1}e^{(h_k - \tau_k)\cos(b(h_k - \tau_k))} \) of \( A \). Here, \( j = 1, 2, \ldots, r \) in case of a real eigenvalue and \( j = 2, 4, \ldots, r \) in case of a pair of complex conjugate eigenvalues, where \( r \) is the size of the largest Jordan block corresponding to the eigenvalue (pair).
The real Jordan form of a matrix $A^{n \times n}$ is a block diagonal matrix $J^{n \times n}$ with the eigenvalues on the diagonal and related to $A$ as
\[ J = Q^{-1}AQ, \tag{2.6} \]
with $Q \in \mathbb{R}^{n \times n}$ a matrix that contains the generalised eigenvectors of $A$. In particular,
\[ J = \text{diag}(J_1, J_2, \ldots, J_p), \tag{2.7} \]
where $J_i \in \mathbb{R}^{n_i \times n_i}$ for all $i = 1, 2, \ldots, p$, with $\sum_{i=1}^{p} n_i = n$, is called a real Jordan block that has a (block) upper triangular form and $p$ denotes the number of distinct eigenvalues. In case of real eigenvalues, each $J_i$ typically has the following structure:
\[ J_i = \begin{bmatrix} \lambda_i & 1 & 0 & \cdots & 0 \\ 0 & \lambda_i & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_i & 1 \\ 0 & 0 & \cdots & 0 & \lambda_i \end{bmatrix}, \tag{2.8} \]
and in case of a pair of complex conjugate eigenvalues, it typically has the following structure:
\[ J_i = \begin{bmatrix} D_i & I & 0 & \cdots & 0 \\ 0 & D_i & I & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & D_i & I \\ 0 & 0 & \cdots & 0 & D_i \end{bmatrix}, \tag{2.9} \]
where $D_i = \begin{bmatrix} a_i & -b_i \\ b_i & a_i \end{bmatrix}$ for the complex eigenvalue pair given by $\lambda_i = a_i \pm b_i \sqrt{-1}$. For more information about the Jordan form and the real Jordan form, the reader is referred to [13, 26] and to Appendix B.1 of [22]. Performing this transformation of the system matrix $A$ to the real Jordan form is the first step in the overapproximation process.

The second step is taken by considering the matrix exponential functions of the continuous-time system matrix $A$ in the discrete-time model in (2.4), i.e. $e^{Ah_k}$ and $e^{As}$. For the matrix exponential $e^{As}$ it holds that
\[ e^{As} = Q \text{diag}(e^{J_1s}, e^{J_2s}, \ldots, e^{J_ps})Q^{-1} = \sum_{l=1}^{\nu} \alpha_{1,l}(s)S_{1,l}, \tag{2.10} \]
where $\alpha_{1,l}(s)$ are varying functions of $s$ and $S_{1,l}$ are constant matrices for all $l = 1, 2, \ldots, \nu$. Now using the real Jordan blocks $J_i$ as in (2.8), the exponential matrix $e^{J_is}$ for real eigenvalues is given by
\[ e^{J_is} = e^{\lambda_is} = \begin{bmatrix} 1 & s & \frac{s^2}{2!} & \cdots & \frac{s^{(r-1)}}{(r-1)!} \\ 0 & 1 & s & \cdots & \frac{s^{(r-2)}}{(r-2)!} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 1 & s \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.11} \]
and $e^{J_is}$ for complex eigenvalues, i.e. with $J_i$ as in (2.9), is given by
\[ e^{J_is} = e^{a_is} = \begin{bmatrix} \tilde{D}_i & s\tilde{D}_i & \frac{s^2}{2!}\tilde{D}_i & \cdots & \frac{s^{(r-1)}}{(r-1)!}\tilde{D}_i \\ 0 & \tilde{D}_i & s\tilde{D}_i & \cdots & \frac{s^{(r-2)}}{(r-2)!}\tilde{D}_i \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \tilde{D}_i & s\tilde{D}_i \\ 0 & 0 & 0 & \tilde{D}_i \end{bmatrix}, \tag{2.12} \]
for the $i$th distinct eigenvalue $\lambda_i$, in which

$$
\tilde{D}_i = \begin{bmatrix}
\cos b_i s & -\sin b_i s \\
\sin b_i s & \cos b_i s
\end{bmatrix}
$$

and where $n_i$ denotes the size of the real Jordan block $J_i$, for $i = 1, 2, \ldots, p$.

In a similar fashion, using the explicit notations of the exponential function in (2.11) and (2.12), the integral of the matrix exponential $e^{As}$ can be determined explicitly as well, i.e.

$$
\int e^{As} ds = Q \text{ diag} \left( \int e^{J_1 s} ds, \int e^{J_2 s} ds, \ldots, \int e^{J_p s} ds \right) Q^{-1}.
$$

(2.13)

For more details about this transformation of the integral terms, the reader is referred to Appendix B.2 of [22], where the explicit expressions are given in case of real and complex eigenvalues.

Now that all exponential and integral terms of the NCS model (2.4) are written as summations of time-varying functions $\alpha_i$ and constant matrices, the entire model can be written as in (2.5), i.e.

$$
\xi_{k+1} = \left( F_0 + \sum_{i=1}^{\zeta} \alpha_i(h_k, \tau_k)F_i \right) \xi_k + \left( G_0 + \sum_{i=1}^{\zeta} \alpha_i(h_k, \tau_k)G_i \right) u_k.
$$

(2.14)

By assuming that the delays and sampling intervals are bounded and contained in the set $\Theta$, as in (2.2), the following set of matrices is obtained

$$
S = \left\{ \left( F_0 + \sum_{i=1}^{\zeta} \alpha_i(h_k, \tau_k)F_i \right), \left( G_0 + \sum_{i=1}^{\zeta} \alpha_i(h_k, \tau_k)G_i \right) \mid (h_k, \tau_k) \in \Theta \right\}.
$$

(2.15)

This set contains an infinite number of matrices (unless $h_{\min} = h_{\max}$ and $\tau_{\min} = \tau_{\max}$), due to the infinite number of values that the uncertain parameters $\tau_k$ and $h_k$ can take. This would make direct linear matrix inequality (LMI) based stability analysis impossible. To arrive at a model that is feasible for analysis, an overapproximated polytopic system with a finite number of vertices is pursued by determining the minimal and maximal values of all functions $\alpha_i$ over $\Theta$, i.e.

$$
\alpha_i = \min_{(h, \tau) \in \Theta} \alpha_i(h, \tau), \quad \overline{\alpha}_i = \max_{(h, \tau) \in \Theta} \alpha_i(h, \tau), \quad i = 1, 2, \ldots, \zeta,
$$

(2.16)

with $\Theta$ as in (2.2). Since each $\alpha_i(h, \tau) \in [\underline{\alpha}_i, \overline{\alpha}_i]$ for all $(h, \tau) \in \Theta$, the set of matrices $S$ is a subset of the convex hull $\text{co}\{H_S\}$, in which

$$
H_S = \left\{ \left( F_0 + \sum_{i=1}^{\zeta} \alpha_i F_i \right), \left( G_0 + \sum_{i=1}^{\zeta} \alpha_i G_i \right) \mid \alpha_i \in [\underline{\alpha}_i, \overline{\alpha}_i], \quad i = 1, 2, \ldots, \zeta \right\}.
$$

(2.17)

Hence, it holds that $S \subseteq \text{co}\{H_S\}$ and the convex hull of $H_S$ has $2^{\zeta}$ vertices.

In the following example, the real Jordan form approach is applied to illustrate the procedure.

**Example 2.1**

Consider the continuous-time system

$$
\dot{x}(t) = Ax(t) + Bu(t),
$$

(2.18)

where

$$
A = \begin{bmatrix}
2 & 1 & 0 \\
0 & 2 & 0 \\
0 & 0 & 5
\end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix}
2 \\
0 \\
1
\end{bmatrix}.
$$
As can be seen, the continuous-time system matrix $A$ is already given in the real Jordan form, which means that $Q = I$, the identity matrix, and the Jordan blocks are given by

\[
J_1 = \begin{bmatrix} 2 & 1 \\ 0 & 2 \end{bmatrix} \quad \text{and} \quad J_2 = 5. \tag{2.19}
\]

The next step is to determine the exponential and integral terms of the model in (2.4) in terms of time-varying functions $\alpha_i$ and constant matrices $F_i$ and $G_i$. Using the procedure as previously discussed (see (2.10) and (2.11)), we write the exponential $e^{As}$ as

\[
e^{As} = Q \text{diag}(e^{J_1s}, e^{J_2s})Q^{-1}, \tag{2.20}
\]

where $Q = I$ and

\[
e^{J_1s} = \begin{bmatrix} e^{2s} & se^{2s} \\ 0 & e^{2s} \end{bmatrix} = e^{2s} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + se^{2s} \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix},
\]

\[
e^{J_2s} = e^{5s},
\]

so that

\[
e^{As} = e^{2s} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} + se^{2s} \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + e^{5s} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.21}
\]

The same procedure is applied on the integral terms, i.e.

\[
\int e^{As}ds = Q \text{diag} \left( \int e^{J_1s}ds, \int e^{J_2s}ds \right)Q^{-1},
\]

where $Q = I$, which yields

\[
\int e^{J_1s}ds = \begin{bmatrix} \frac{1}{5}e^{2s} & \frac{1}{2}se^{2s} - \frac{1}{4}e^{2s} \\ 0 & \frac{1}{2}e^{2s} - \frac{1}{4}e^{2s} \end{bmatrix} = e^{2s} \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{1}{2} \end{bmatrix} + se^{2s} \begin{bmatrix} 0 & \frac{1}{2} \\ 0 & 0 \end{bmatrix},
\]

\[
\int e^{J_2s}ds = \frac{1}{5}e^{5s}.
\]

Applying the appropriate integration limits to these expressions and multiplying the constant matrices with $B$, results in the following time-varying functions and constant matrices that form the representation in (2.5):

\[
\alpha_1(h_k) = e^{2h_k}, \quad \alpha_2(h_k) = h_k e^{2h_k}, \quad \alpha_3(h_k) = e^{5h_k}, \quad \alpha_4(h_k, \tau_k) = e^{2(h_k - \tau_k)},
\]

\[
\alpha_5(h_k, \tau_k) = (h_k - \tau_k) e^{2(h_k - \tau_k)}, \quad \alpha_6(h_k, \tau_k) = e^{5(h_k - \tau_k)}, \tag{2.23}
\]

\[
F_0 = 0, \quad F_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad F_2 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad F_3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & \frac{1}{5} \end{bmatrix},
\]

\[
F_4 = \begin{bmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad F_5 = 0, \quad F_6 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{5} \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{2.24}
\]

Moreover, $G_0 = [-1 \quad 0 \quad -1/5 \quad 1]^T$ and the constant matrices $G_i$, for $i = 1, 2, \ldots, 6$, are equal to

\[
G_1 = 0, \quad G_2 = 0, \quad G_3 = 0, \quad G_4 = [1 \quad 0 \quad 0 \quad 0]^T, \quad G_5 = 0, \quad G_6 = [0 \quad 0 \quad 1/5 \quad 0]^T. \tag{2.25}
\]

With this model, a convex hull is constructed by determining the minimal and maximal values of all functions $\alpha_i$, as in (2.16). In this example, including time-varying sampling intervals $h_k$ and time-varying delays $\tau_k$, the number $\zeta = 2\nu$ of time-varying functions $\alpha_i$ is equal to six, which results in a convex hull that consists of $2^5 = 32^6 = 64$ vertices. This matrix polytope, with its 64 vertices, can be used to analyse stability, as discussed in Section 2.2.
2.1.2 Cayley-Hamilton Approach

Another way to make an overapproximation of the NCS model in (2.4) is to apply the approach based on the Cayley-Hamilton theorem (see, e.g., [11]).

**Theorem 1** Let \( A \in \mathbb{R}^{n \times n} \) be a given matrix and \( I_n \) the identity matrix of dimensions \( n \times n \). Then the characteristic polynomial of \( A \) is defined as \( p(\lambda) := \det(\lambda I_n - A) \). Substituting matrix \( A \) in the characteristic polynomial results in \( p(A) = 0 \).

See [2] for the proof of this theorem. We will use the minimal polynomial of \( A \) that satisfies \( p_{\text{min}}(A) = 0 \) as well. Based on the Cayley-Hamilton theorem, the degree of the minimal polynomial (denoted by \( \nu \)) is smaller than or equal to \( n \). Actually, we have \( \nu = n \) when each distinct eigenvalue of \( A \) corresponds to one Jordan block only and \( \nu < n \) otherwise. Using this minimal polynomial it is possible to express all powers of \( A \) of order \( \nu \) and higher as a combination of the first \( \nu \) powers of \( A \), i.e.

\[
A^i = c_{i,0}I + c_{i,1}A + c_{i,2}A^2 + \ldots + c_{i,\nu-1}A^{\nu-1} = \sum_{j=0}^{\nu-1} c_{i,j}A^j, \tag{2.26}
\]

for some \( i \in \mathbb{N}_{\geq \nu} \) and where the coefficients \( c_{i,j} \) are constants for \( j = 0, 1, \ldots, \nu - 1 \). Furthermore, for all \( i = 0, 1, \ldots, \nu - 1 \) and \( j = 0, 1, \ldots, \nu - 1 \), let \( c_{i,j} = 1 \) when \( i = j \) and \( c_{i,j} = 0 \) when \( i \neq j \).

Since the matrix exponential of the product \( AS \) is written as

\[
e^{As} = \sum_{i=0}^{\infty} \frac{A^i s^i}{i!}, \tag{2.27}
\]

and \( A^i \) can be written as in (2.26), we now have that

\[
e^{As} = \sum_{i=0}^{\infty} \frac{\sum_{j=0}^{\nu-1} c_{i,j}A^j s^i}{i!}, \tag{2.28}
\]

and therefore

\[
e^{As} = \sum_{j=1}^{\nu} f_j(s)A^{j-1}, \tag{2.29}
\]

where \( f_j(s) = \sum_{i=0}^{\infty} \frac{c_{i,j}s^i}{i!}, \ j = 0, 1, \ldots, \nu - 1, \) are uncertain functions that depend on \( s \). The functions \( f_j \) can be approximated by taking a high-order truncation, e.g., by \( f_j(s) \approx \sum_{i=0}^{m} \frac{c_{i,j}s^i}{i!} \) and \( m \gg 1 \), as is done in [11]. Another way is to determine these varying functions \( f_j \) exactly.

To do so, combinations of the varying functions \( \alpha_l \) that result from the matrix exponential \( e^{As} \) in (2.10) based on the real Jordan form approach as discussed in Section 2.1.1 (see (2.21) for typical forms of the functions \( \alpha_l \) are used, i.e.

\[
f_{1,j} = \sum_{l=1}^{\nu} \beta_{1,j,l}\alpha_{1,l}(s), \tag{2.30}
\]

with constants \( \beta_{1,j,l} \) for \( j, l \in \{1, 2, \ldots, \nu\} \) that can be computed using

\[
\sum_{j=1}^{\nu} \beta_{1,j,l}A^{j-1} = QS_{1,l}Q^{-1}, \tag{2.31}
\]

where

\[
\sum_{l=1}^{\nu} \alpha_{1,l}S_{1,l} = e^{As}, \tag{2.32}
\]

which can be computed using the overapproximation approach based on the real Jordan form.
Hence, by determining the constants $\beta_{1,j,t}$ using (2.31) and $\alpha_{1,t}$ using (2.32), the varying functions $f_{1,j}$ can be determined. In a similar fashion, we have that

$$f_{2,j} = \sum_{l=1}^{2\nu} \alpha_{2,j,l}\alpha_{1,t} \quad \text{and} \quad f_{3,j} = \sum_{l=1}^{\nu} \beta_{3,j,l}\alpha_{1,t}, \quad (2.33)$$

for $j = 1, 2, \ldots, \nu$, where $\beta_{2,j,l}$ and $\beta_{3,j,l}$ are computed using

$$\sum_{j=1}^{\nu} \beta_{1,j,t}A^{j-1} = QS_{i,t}Q^{-1}, \quad \text{for } i = 2, 3, \quad (2.34)$$

and

$$\sum_{l=1}^{2\nu} \alpha_{2,l}S_{2,l} = \int_{h_k - \tau_k}^{h_k} e^{A_1 s} ds \quad \text{and} \quad \sum_{l=1}^{\nu} \alpha_{3,l}S_{3,l} = \int_{0}^{h_k - \tau_k} e^{A_1 s} ds. \quad (2.35)$$

This results in the following system representation

$$\xi_{k+1} = \left[ \begin{array}{c} \sum_{j=1}^{\nu} f_{1,j}(h_k)A^{j-1} \\ 0 \end{array} \right] \xi_k + \left[ \begin{array}{c} S_{3,0} + \sum_{j=1}^{\nu} f_{3,j}(h_k, \tau_k)A^{j-1}B \\ I \end{array} \right] u_k, \quad (2.36)$$

where $S_{3,0}$ is a constant matrix and $f_{1,j}(h_k)$, $f_{2,j}(h_k, \tau_k)$ and $f_{3,j}(h_k, \tau_k)$ are varying functions that are combinations of the functions $\alpha_{1,t}$ for $i = 2, 3$. Note that all functions $f_{2,j}$ are equal to $f_{1,j}$ or $f_{3,j}$ for all $j = 1, 2, \ldots, \nu$, due to the integration bounds $h_k - \tau_k$ and $h_k$ of the corresponding integral that result in functions that are also contained in the exponential and the other integral term.

By assuming that the delays and the sampling intervals in model (2.36) are bounded and contained in the set $\Theta$, as in (2.2), the following set of matrices is obtained

$$\mathcal{S} = \left\{ \left( F_0 + \sum_{j=1}^{\nu} \alpha_j(h_k, \tau_k)F_j \right), \left( G_0 + \sum_{i=1}^{2\nu} \alpha_j(h_k, \tau_k)G_j \right) \mid (h_k, \tau_k) \in \Theta \right\}, \quad (2.37)$$

where $F_0 = 0$, $G_0 = S_{3,0}$, $\zeta = 2\nu$ and

$$\alpha_j = \begin{cases} f_{1,j} & \text{for } j = 1, 2, \ldots, \nu, \\ f_{3,j-\nu} & \text{for } j = \nu + 1, \nu + 2, \ldots, 2\nu. \end{cases} \quad (2.38)$$

This set $\mathcal{S}$ contains an infinite number of matrices, due to the infinite number of values for $h_k$ and $\tau_k$. Similar to Section 2.1.1, the system is overapproximated by taking all combinations of the minimal and maximal values of $\alpha_j$, i.e.

$$\underline{\alpha}_j = \min_{(h_k, \tau_k) \in \Theta} \alpha_j, \quad \overline{\alpha}_j = \max_{(h_k, \tau_k) \in \Theta} \alpha_j, \quad (2.39)$$

for $j = 1, 2, \ldots, 2\nu$ and $\Theta$ is defined in (2.2). Using this, we obtain a polytopic overapproximation with $2^{2\nu} \alpha_j$ vertices, i.e.

$$\mathcal{S} \subseteq \text{co}\{\mathcal{H}_\mathcal{S}\}, \quad (2.40)$$

where

$$\mathcal{H}_\mathcal{S} = \left\{ \left( F_0 + \sum_{j=1}^{\nu} \alpha_jF_j \right), \left( G_0 + \sum_{j=1}^{\nu} \alpha_jG_j \right) \mid \alpha_j \in \{\underline{\alpha}_j, \overline{\alpha}_j\}, \quad j = 1, 2, \ldots, \zeta \right\}. \quad (2.41)$$

Observe that set $\mathcal{H}_\mathcal{S}$ contains all $2^{2\nu} \alpha_j$ combinations of the time-varying functions $\alpha_j$. This number of vertices can be reduced to $2 \cdot 2\nu$, by applying Theorem 2 from [11]. However, this technique is not taken into account during the comparison later on in this thesis, but is implemented in the Matlab toolbox.

To illustrate this overapproximation procedure, the Cayley-Hamilton theorem based approach is applied to an example.
Example 2.2

Reconsider the system as in Example 2.1, i.e.

\[
\dot{x}(t) = Ax(t) + Bu(t),
\]  
(2.42)

where

\[
A = \begin{bmatrix} 2 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 5 \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} 2 \\ 0 \\ 1 \end{bmatrix}.
\]

Rewriting this system in terms of varying functions and constant matrices using the real Jordan form approach as in Example 2.1, results in

\[
S_{1,l} = \sum_{j=1}^{\nu} \beta_{1,j,l} J^{j-1},
\]  
(2.44)

for \( l = 1, 2, 3 \). Using this equation and the constant matrices \( S_{1,l} \) from (2.43), results in the following constants \( \beta_{1,j,l} \) for all \( j, l \in \{1, 2, 3\} \):

\[
\beta_{1,1,1} = \frac{5}{9}, \quad \beta_{1,1,2} = -\frac{10}{3}, \quad \beta_{1,1,3} = \frac{4}{9}, \quad \beta_{1,2,1} = \frac{4}{9}, \quad \beta_{1,2,2} = \frac{7}{3}, \\
\beta_{1,2,3} = -\frac{4}{3}, \quad \beta_{1,3,1} = \frac{1}{9}, \quad \beta_{1,3,2} = -\frac{1}{3}, \quad \beta_{1,3,3} = \frac{1}{9}.
\]  
(2.45)

With these constants and the varying functions \( \alpha_{1,l} \) from (2.43), the varying functions \( f_{i,j} \) can be computed using (2.30), i.e.

\[
f_{1,1} = \frac{5}{9} e^{2s} - \frac{10}{3} e^{2s} + \frac{4}{9} e^{5s},
\]

\[
f_{1,2} = \frac{4}{9} e^{2s} + \frac{7}{3} e^{2s} - \frac{4}{9} e^{5s},
\]

\[
f_{1,3} = -\frac{1}{9} e^{2s} - \frac{1}{3} e^{2s} + \frac{1}{9} e^{5s}.
\]  
(2.46)

The same transformation is applied on all exponential and integral terms of the NCS model (2.4), such that all terms are written as summations of varying functions \( f_{i,j} \), for all \( i, j \in \{1, 2, 3\} \), and powers of the system matrix \( A \). Each of the functions \( f_{i,j} \) results to be a combination of some of the following uncertain terms:

\[
e^{2h_k}, \quad h_k e^{2h_k}, \quad e^{5h_k}, \quad e^{2(h_k - \tau_k)}, \quad (h_k - \tau_k)e^{2(h_k - \tau_k)}, \quad e^{5(h_k - \tau_k)}.
\]  
(2.47)

Now, the minimal and maximal values of the terms in (2.47) are determined according to (2.39) and with the obtained values we construct the minimal and maximal values of the functions \( \alpha_j \) as in (2.38), such that the vertices of the overapproximation are obtained in the set \( \mathcal{H}_S \) as in (2.41), which has \( 2^{2^6} = 2^6 = 64 \) vertices. This matrix polytope is used to analyse stability, as will be discussed next.
2.2 Stability Analysis Using Lyapunov Functions

After constructing a polytopic overapproximation of the NCS model, it is possible to analyze stability of the resulting polytopic system. Using particular types of quadratic Lyapunov functions, stability conditions are obtained in terms of linear matrix inequalities (LMIs), which can be solved efficiently.

In this thesis, two types of Lyapunov functions (LFs) are considered, namely

- common quadratic Lyapunov function (CQLF) and
- parameter-dependent quadratic Lyapunov function (PDLF).

Both types of LFs are discussed by using the overapproximations $H_S$ from the previous sections. However, to analyse the stability of the closed-loop NCS, a control law is needed. In this thesis, we use a static state feedback of the form $u_k = -Kx_k$, which results in a lifted controller of the form $u_k = -[K \ 0]\mathbf{x}_k$. If we apply the state feedback to the overapproximated open-loop models in (2.17) and (2.41), the following set of matrices is obtained:

\[ H_S = \left\{ \left( S_0 + \sum_{i=1}^{\zeta} \alpha_i S_i \right) \mid \alpha_i \in \{ \alpha_i, \overline{\alpha}_i \}, \ i = 1, 2, \ldots, \zeta \right\}, \]  

(2.48)

where $\zeta = 2\nu$. On this set $H_S$ a robust stability analysis is performed using Lyapunov functions.

Now, we discuss both types of Lyapunov functions (LFs) by considering the overapproximated closed-loop NCS model, i.e.

\[ \mathbf{x}_{k+1} = \mathcal{H}_i \mathbf{x}_k, \]  

(2.49)

where the matrix $\mathcal{H}_i$ belongs to the polytopic set $H_S$ as in (2.48), i.e. $\mathcal{H}_i \subseteq \text{co}\{H_S\}$. Moreover, it holds that

\[ \mathcal{H}_i = \sum_{i=1}^{2\zeta} \rho_i(h_k, \tau_k)S_i, \text{ for } \rho_i(h_k, \tau_k) \geq 0, \sum_{i=1}^{2\zeta} \rho_i(h_k, \tau_k) = 1, \]  

(2.50)

where $S_i$ denote the vertices of the set $H_S$, i.e. $H_S = \{S_1, S_2, \ldots, S_{2\zeta}\}$. To analyse this overapproximated system on stability, we consider a PDLF of the form

\[ V(\mathbf{x}_k) = \xi_k^T \sum_{i=1}^{2\zeta} \rho_i(h_k, \tau_k) P_i \xi_k, \]  

(2.51)

with $P_i$, for $i = 1, 2, \ldots, 2\zeta$, are symmetric and positive definite matrices. The closed-loop NCS is asymptotically stable if there exist such a LF (2.51) satisfying

\[ \xi_k^T P_i \xi_k > 0, \]

\[ \xi_{k+1}^T P_i \xi_{k+1} - \xi_k^T P_j \xi_k < -\gamma \xi_k^T P_j \xi_k, \]  

(2.52)

for all $\xi_k \neq 0$ and for all $i, j \in \{1, 2, \ldots, N\}$. Moreover, in (2.52) the parameter $0 \leq \gamma < 1$ ensures that the LF decreases with rate $1 - \gamma$. See [5, 6] for more details about parameter-dependent quadratic Lyapunov functions (PDLFs). Note that, when $P_i = P_j = P$ the LF is called common quadratic as one matrix $P$ needs to satisfy all conditions for all $\xi_k \neq 0$ in order to have a globally asymptotically stable system.

To transform the Lyapunov conditions (2.52) into LMIs, the matrices $\mathcal{H}_j$ of the overapproximated NCS model (2.49) are inserted in (2.52), i.e.

\[ \xi_k^T \left( \mathcal{H}_j^T P_i \mathcal{H}_j - (1 - \gamma) P_j \right) \xi_k < 0, \]  

(2.53)
which results in the following set of LMIs

\[ P_i > 0, \]
\[ \mathcal{H}_j^T P_j \mathcal{H}_j - (1 - \gamma) P_j < 0, \quad (2.54) \]

for \( i, j \in \{1, 2, \ldots, 2^\zeta \} \), where \( P_i = P_j = P \) in case a common quadratic LF is desired. Now, if matrices \( P_i \) exist that satisfy all LMIs in (2.54), the overapproximated NCS (and as a consequence the NCS (2.4) with controller \( u_k = -Kx_k \)) is globally asymptotically stable.

**2.3 Controller Synthesis**

In the previous section, we assumed that the state feedback is available, i.e. matrix \( K \) of the control law \( u_k = -Kx_k \) is known. However, it is possible to perform a controller synthesis when the controller \( K \) is not available by assuming that the state feedback is based on the augmented state vector \( \xi_k \), i.e.

\[ u_k = -\bar{K}\xi_k. \quad (2.55) \]

As the augmented state vector equals \( \xi_k = [x_k^T \ u_{k-1}^T]^T \), the controller (2.55) can be written as

\[ u_k = -\bar{K}x_k - \bar{K}u_{k-1}. \quad (2.56) \]

However, a control law that depends on the previous controller input \( u_{k-1} \) might result in problems when dropouts occur in the NCS (see, e.g., [3]). For this reason, we choose to use a state feedback \( u_k = -Kx_k \) that only depends on the state vector \( x_k \), i.e. \( \bar{K} = [K \ 0_{m \times m}] \). To find such a \( K \), controller synthesis can be applied to the overapproximated NCS model using common quadratic or parameter-dependent Lyapunov functions.

When the exact model (2.4) is overapproximated (without applying a control law to the model), we obtain the finite set of matrices as in (2.17) and (2.41), i.e.

\[ \mathcal{H}_{FG} = \left\{ \left( F_0 + \sum_{i=1}^{\zeta} \alpha_i F_i, \ (G_0 + \sum_{i=1}^{\zeta} \alpha_i G_i) \right) \mid \alpha_i \in \{\alpha, \bar{\alpha}\}, \ i = 1, 2, \ldots, \zeta \right\}, \quad (2.57) \]

The set of vertices \( \mathcal{H}_{FG} \) is also written as \( \mathcal{H}_{FG} = \{H_{F,j}, H_{G,j} \mid j = 1, 2, \ldots, 2^\zeta \} \). With this finite set of matrices, a finite number of LMI controller synthesis conditions are given for the state feedback controller \( u_k = -Kx_k \) in the following theorem that uses a parameter-dependent Lyapunov function.

**Theorem 2** Consider the NCS model (2.1) and its discrete-time representation (2.4) for sequences of sampling instants and delays as in (2.2). Consider the equivalent representation (2.5) based on the real Jordan form (or (2.36) based on the Cayley-Hamilton theorem) and the set of vertices \( \mathcal{H}_{FG} \) defined in (2.57).

If there exist symmetric positive definite matrices \( Y_j \in \mathbb{R}^{(n+m) \times (n+m)} \), a matrix \( \bar{Z} \in \mathbb{R}^{m \times n} \), matrices \( X_j = \begin{bmatrix} X_1 & 0 \\ X_{2,j} \ X_{3,j} \end{bmatrix} \), with \( X_1 \in \mathbb{R}^{n \times n}, X_{2,j} \in \mathbb{R}^{m \times n}, X_{3,j} \in \mathbb{R}^{m \times m}, j = 1, 2, \ldots, 2^\zeta \), and a scalar \( 0 \leq \gamma < 1 \) that satisfy

\[ \begin{bmatrix} X_j + X_j^T - Y_j & X_j^T H_{F,j} - [\bar{Z} \ 0]^T \ (1 - \gamma) Y_{f_{j}} \\ H_{F,j} X_j - H_{G,j} \ [\bar{Z} \ 0] & H_{G,j}^T \ (1 - \gamma) Y_{f_{j}} \end{bmatrix} > 0, \quad (2.58) \]

for all \( j, l \in \{1, 2, \ldots, 2^\zeta \} \), then the closed-loop NCS (2.1) with \( K = \bar{Z}X_1^{-1} \) is globally asymptotically stable.

See [3] for a proof of this theorem. If, in this theorem, we set \( Y_j = Y_f = Y \), the result is a common quadratic LF based controller synthesis.
Chapter 3

Comparison of Dropout Modelling Approaches

In the previous chapter, an NCS model is considered that incorporates two network phenomena, namely time-varying sampling intervals $h_k$ and time-varying delays $\tau_k$. In this chapter, we consider an NCS model that also incorporates packet dropouts. Moreover, the focus of this chapter is on the different ways of modelling these dropouts.

This chapter is based on a paper that is submitted to the 2010 American Control Conference, Baltimore, Maryland, USA. As a consequence, there might be some overlap with the previous chapter.
On Dropout Modelling for Stability Analysis of Networked Control Systems\textsuperscript{1}

J.J.C. van Schendel, M.C.F. Donkers, W.P.M.H. Heemels, N. van de Wouw

Abstract—This paper presents three discrete-time modelling approaches for networked control systems (NCSs) that incorporate time-varying sampling intervals, time-varying delays and dropouts. The focus of this work is on the extension and comparison of two existing techniques to describe dropouts, namely (i) dropouts modelled as prolongation of the delay and (ii) dropouts modelled as prolongation of the sampling interval, and a new approach (iii) based on explicit dropout modelling using automata. Based on polytopic overapproximations of the resulting discrete-time NCS models, we provide LMI-based stability conditions. Herewith, we compare the extensions of the existing approaches and the newly proposed method in terms of modelling accuracy, conservatism and complexity of the stability analysis. Using an illustrative example, we provide a thorough numerical comparison of the alternative modelling approaches.

3.1 Introduction

The literature on modelling, analysis and controller design of networked control systems (NCSs) expanded rapidly over the last decade, see, e.g., [1, 25, 33]. The reason for the interest in the research area is that the use of networks offers many advantages, such as low installation and maintenance costs, reduced system wiring (in the case of wireless networks) and increased flexibility of the system. However, from a control theory point of view, the presence of the network also introduces several disadvantages such as time-varying networked-induced delays, aperiodic sampling and/or packet dropouts.

Before deploying NCS in industrial environments, a deep understanding of the effects of packet dropouts, time-varying sampling intervals and time-varying delays on the stability and performance of the NCS is needed. Most of the literature that deals with stability and stabilisation of NCSs focuses only on one of these phenomena, while ignoring the others. Clearly, it is important to consider the combined presence of dropouts, time-varying sampling intervals and time-varying delays, as in any practical NCS these typically occur simultaneously. Some results are available in the literature that consider at least two of these network phenomena. In [28, 29, 34] stability and stabilisation of NCSs with packet dropouts and delays are investigated, based on discrete-time NCS representations. However, they assume that the delay is constant, which is often not realistic. In [16], the stability and disturbance attenuation of a NCS with time-varying delays and packet dropouts are investigated, based on a switched system approach. Therein, it is assumed that both the controller and sensor act in a time-driven periodic manner resulting in the fact that the delays take values in a finite set only, which is upper-bounded by the sampling interval.

Continuous-time modelling approaches, based on (impulsive) delay-differential equations, including packet dropouts and time-varying delays, are described in [30, 32, 19]. For these models, both stability analysis and controller synthesis methods are proposed and time-varying delays both smaller and larger than the sampling interval are included. Recently, in [4] a discrete-time approach was proposed for stability analysis and state feedback synthesis for NCS with both delays.

\textsuperscript{1}This work is partially supported by the European Community through the FP7-ICT-2007-2 thematic programme under the WIDE-224168 project.

Jos van Schendel, Tijs Donkers, Maurice Heemels and Nathan van de Wouw are with the Mechanical Engineering department of Eindhoven University of Technology, P.O.Box 513, 5600 MB Eindhoven, The Netherlands.

j.j.c.v.schendel@student.tue.nl, {m.c.f.donkers, m.heemels, n.v.d.wouw}@tue.nl
and dropouts. Typically, in the latter approaches based on either discrete-time or continuous-time models the inclusion of packet dropouts is modelled as prolongation of the maximal delay or the maximal sampling interval. As this does not truly model the effect of dropouts and introduces spurious solutions, these approaches have some degree of conservatism in the sense that the models allow for sequences of control updates which can not occur in real NCSs. In [10], also the problem of stability analysis of NCS with dropouts and variable delays is considered based on a discrete-time modelling approach. The occurrence of dropouts is modelled as prolongation of the sampling interval, which also leads to spurious solutions, and in the stability analysis a common quadratic Lyapunov function (LF) approach is adopted, which is conservative when compared to a parameter-dependent LF. In addition, the effect of varying sampling intervals and a systematic method for transforming the infinite set of linear matrix inequalities (LMIs) (due to the infinite number of possibilities for the delays) into a finite set are missing in [10].

This paper will provide a general stability analysis method based on a discrete-time modelling approach for NCS including all the three mentioned network phenomena: packet dropouts, varying sampling intervals and varying delays. In particular, we provide three different models that each accommodate for packet dropouts in a different way. In doing so, the above mentioned drawbacks of existing approaches can be studied. We will study NCSs in which the delays are time-varying and can take values from a bounded set, containing an infinite number of values (i.e. \( \tau_k \in [\tau_{\min}, \tau_{\max}] \)), time-varying sampling intervals \( h_k \) taking values in \( [h_{\min}, h_{\max}] \) and dropouts for which we only bound the maximal number of successive dropouts by \( \bar{\delta} \). By adopting a discrete-time hybrid automaton modelling approach (see, e.g., [27, 18, 7]), we truly describe the effect of packet dropouts instead of incorporating it (artificially) using prolongations of the maximal delay or maximal sampling interval as done in the literature before. We compare this newly proposed explicit approach with two (adapted) methods from [4] and [10]. These adaptations consider small delays, time-varying sampling intervals and mode-dependent (i.e. dropout-dependent) Lyapunov functions (LFs), which lead to LMI-based conditions for stability of the NCS that alleviate the conservatism in the existing approaches.

The remainder of this paper is organised as follows: In Section 3.2, we introduce two NCS models from the literature, and extend them, and propose a novel way to model packet dropouts. In Section 3.3, we introduce our polytopic overapproximation method. Based on an overapproximated model, we provide conditions for stability in terms of LMIs in Section 3.4. The drawbacks of the existing approaches and the motivations of the extensions are discussed in Section 3.5. Finally, we illustrate the results using a numerical example in Section 3.6 and present concluding remarks in Section 3.7.

### 3.2 The NCS Model

In this section, the discrete-time description of a NCS including the unknown and time-varying delays, unknown and time-varying sampling intervals and packet dropouts is presented. The NCS is depicted schematically in Fig. 3.1. It consists of a linear continuous-time plant

\[
\dot{x}(t) = Ax(t) + Bu^*(t),
\]

with \( A \in \mathbb{R}^{n \times n} \) and \( B \in \mathbb{R}^{n \times m} \), and a discrete-time static time-invariant controller, which are connected over a communication network that induces network delays (namely the sensor-to-controller delay \( \tau_{sc} \) and the controller-to-actuator delay \( \tau_{ca} \)). The state measurements are sampled at the sampling time instants \( s_k \):

\[
s_k = \sum_{i=0}^{k-1} h_i \quad \forall k \geq 1, \quad s_0 = 0,
\]

which are non-equidistantly spaced in time due to the time-varying sampling intervals \( h_k > 0 \). The sequence of sampling instants \( s_0, s_1, s_2, \ldots \) is strictly increasing in the sense that \( s_{k+1} > s_k \), for all \( k \in \mathbb{N} \). We denote by \( x_k := x(s_k) \) the \( k \)th sampled value of \( x \) and by \( u_k \) the corresponding
control value. Packet dropouts may occur (see Fig. 3.1) and are modelled by the parameter $m_k$. This parameter denotes whether or not a packet is dropped:

$$m_k = \begin{cases} 
0, & \text{if } x_k \text{ and } u_k \text{ are received}, \\
1, & \text{if } x_k \text{ and/or } u_k \text{ is dropped}.
\end{cases} \quad (3.3)$$

In (3.3), we make no distinction between packet dropouts that occur in the sensor-to-controller connection and the controller-to-actuator connection in the network. This can be justified by realising that, for static controllers, the effect of the packet dropouts on the control updates implemented on the plant is the same in both cases. Indeed, for packet dropouts between the sensor and the controller no new control update is computed and thus no new control input is sent to the actuator. In the case of packet dropouts between the controller and the actuator no new control update is received by the actuator either. Finally, the zero-order-hold (ZOH) function (in Fig. 3.1) is applied to transform the discrete-time control input $u_k$ to a continuous-time control input $u^*(t)$ being the actual actuation signal of the plant.

In the model, both the varying computation time ($\tau_c^k$), needed to evaluate the controller, and the varying network-induced delays, i.e. the sensor-to-controller delay ($\tau_{sc}^k$) and the controller-to-actuator delay ($\tau_{ca}^k$), are taken into account. We assume that the sensor acts in a time-driven fashion (i.e. sampling occurs at the times $s_k$ defined in (3.2)) and that both the controller and the actuator act in an event-driven fashion (i.e. responding instantaneously to newly arrived data). Under these assumptions, all three delays can be captured by a single delay $\tau_k^* := \tau_{sc}^k + \tau_c^k + \tau_{ca}^k$, see also [33] and [20]. To include these effects in the continuous-time model, let us define the parameter $k^*(t)$ that denotes the index of the most recent control input that is available at time $t$ as $k^*(t) := \max\{k \in \mathbb{N} | s_k + \tau_k \leq t \land m_k = 0\}$. The continuous-time model of the plant of the NCS is then given by

$$\dot{x}(t) = Ax(t) + Bu^*(t), \quad u^*(t) = u_{k^*(t)}, \quad (3.4)$$

with $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{n \times m}$. Here, we assume that the most recent control input remains active in the plant if a packet is dropped.

We assume that the delays are bounded and contained in the set $[\tau_{\min}, \tau_{\max}]$, with $\tau_{\max} \geq \tau_{\min} \geq 0$, the sampling intervals are also bounded and lie in the set $[h_{\min}, h_{\max}]$, with $h_{\max} \geq h_{\min}$, and that delays are equal or smaller than the sampling intervals $\tau_k \leq h_k$, for all $k \in \mathbb{N}$. Therefore, we have that $(h_k, \tau_k) \in \Theta$, for all $k \in \mathbb{N}$, where

$$\Theta = \left\{ (h, \tau) \in \mathbb{R}^2 \mid h \in [h_{\min}, h_{\max}], \tau \in [\tau_{\min}, \min\{h, \tau_{\max}\}] \right\}. \quad (3.5)$$

Furthermore, the number of subsequent packet dropouts is upper bounded by $\bar{\delta}$. This means that

$$\sum_{v=k-\bar{\delta}}^{k} m_v \leq \bar{\delta}, \quad (3.6)$$

and guarantees that from the sequence of control inputs $u_{k-\bar{\delta}}, u_{k-\bar{\delta}+1}, \ldots, u_k$ at least one is implemented.
In this work, we consider three different approaches for modelling dropouts in which we use an exactly discretised version of the model in (3.4) that includes varying delays, varying sampling intervals and packet dropouts: (i) dropouts modelled as prolongation of the delay (see, e.g., [4, 3]), (ii) dropouts modelled as prolongation of the sampling interval (see, e.g., [10]) and (iii) a novel explicit dropout modelling approach. The essence of the three methods is presented schematically in Fig. 3.2. In this figure, a circle represents a sample that is not dropped and a cross a sample that is dropped. As shown, the discrete-time NCS model in methods (i) and (iii) describe the state evolution from $s_k$ to $s_{k+1}$, while method (ii) discards the dropped samples and does not include the corresponding states $x(s_k)$ in the discrete-time model. This results in prolonged sampling intervals in case of method (ii). Method (i) includes the drops through prolongation of the delay using the parameters $t_j$ for $j = 0, \ldots, \delta + 2$, which indicate the control update times, see [3]. Method (iii) keeps track of dropouts using an explicit automaton model. We describe these three methods in more detail in the next section. In addition, we present an extension to the existing approach of [10] in section 3.2.2.

Before presenting the three techniques, let us first introduce the variable $\delta_k$, which denotes the number of subsequent dropouts at sample instant $s_k$, i.e. $\delta_k = \delta$ means that $u_{k-1}, u_{k-2}, \ldots, u_{k-\delta}$ are dropped and $u_{k-\delta-1}$ is received at the plant. Note that $\delta_k \in \{0, 1, \ldots, \delta\}$. Given the fact that $m_k \in \{0, 1\}$ (see (3.3)), we can write the evolution of $\delta_k$ as

$$
\delta_{k+1} = m_k(\delta_k + 1). \tag{3.7}
$$

Equation (3.7) expresses that $\delta_{k+1}$ is increased by one if at the $k$th sampling instant the packet dropped and is reset to zero if the packet arrived. The variable $\delta_k$ will be used in the second and third modelling method that are described next.

### 3.2.1 Dropouts as Prolongation of Delays

In [3], NCSs are studied including delays that are larger than the sampling intervals ($\tau_k > h_k$) and dropouts, which are modelled as prolongations of the transmission delays. For the sake of simplicity, this approach is adapted to the case where delays are smaller than the sampling intervals ($\tau_k \leq h_k$) here. The NCS model (3.4) including the assumptions on the sampling intervals, delays and dropouts in (3.5) and (3.6) can be captured in a discrete-time representation, based on an exact discretisation of (3.4) at the sampling instants $\{s_k\}_{k \in \mathbb{N}}$. From $s_k$ to $s_{k+1}$ this yields

$$
x_{k+1} = e^{Ah_k}x_k + \sum_{j=0}^{\delta+1} \int_{h_{k-1}^j}^{h_{k-1}^{j+1}} e^{As}Bu_{k+j-\delta-1}ds, \tag{3.8}
$$

where $x_k := x(s_k)$ is the discrete-time state at the $k$th sampling instant $s_k$ and $h_k \geq t_j \geq 0$, for all $j \in \{0, 1, \ldots, \delta + 1\}$, represent the control update instants in the sampling interval $[s_k, s_{k+1}]$. 

![Figure 3.2: Different dropout modelling approaches.](image-url)
in the sense that \( u_{k+j-\delta-1} \) is active in \([s_k + t^j_k, s_k + t^k_{j+1})\). See [3] for more details. Applying a static state feedback controller \( u_k = -\hat{K} x_k \) to system (3.8), defining the augmented state vector by
\[
\xi_k := \left[ x_k^T \ u_{k-1}^T \ \ldots \ u_{k-\delta-1}^T \right]^T
\]
and introducing the vector \( \rho_k \) of uncertain parameters consisting of the sampling and the actuation update instants \( \rho_k := (h_k, t^k_1, \ldots, t^k_\delta) \), results in the following discrete-time NCS model
\[
\xi_{k+1} = \tilde{A}_1(\rho_k)\xi_k,
\] (3.9)
where
\[
\tilde{A}_1(\rho_k) = \begin{bmatrix}
\Lambda(\rho_k) & M_\delta(\rho_k) & M_{\delta-1}(\rho_k) & \ldots & M_0(\rho_k) \\
-\hat{K} & 0 & 0 & \ldots & 0 \\
0 & I & 0 & \ldots & 0 \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
0 & \ldots & 0 & I & 0
\end{bmatrix}
\] (3.10)

In (3.10), \( \Lambda(\rho_k) = e^{Ah_k} - M_{\delta+1}(\rho_k)\hat{K} \) and
\[
M_j(\rho_k) = \int_{h_k-t^j_{j+1}}^{h_k-t^j_k} e^{As}dB,
\] (3.11)
for all \( j \in \{0, 1, \ldots, \delta + 1\} \). The actuation instants \( t^j_k \) lie in the set \([t^k_{j,\min}, t^k_{j,\max}]\), where \( t^k_{j,\min} \) and \( t^k_{j,\max} \) are defined as
\[
t^k_{j,\min} = \begin{cases} 
\tau_{\min} & \text{if } j = \delta + 1, \\
0 & \text{if } 1 \leq j < \delta + 1,
\end{cases}
\] (3.12)
and
\[
t^k_{j,\max} = \begin{cases} 
h_{\max} & \text{if } 1 < j \leq \delta + 1, \\
\tau_{\max} & \text{if } j = 1.
\end{cases}
\] (3.13)
Additionally, \( t^k_0 := 0 \) and \( t^k_{\delta+2} := h_k \). See [4, 3] for the proof of the validity of the bounds on \( t^k_j \).

### 3.2.2 Dropouts as Prolongation of Sampling Intervals

In [10], García-Rivera and Barreiro model dropouts as prolongations of the constant sampling intervals \( h \), by using the dropout counter \( \delta_k \) from (3.7). Basically, the idea is that a sampling instant \( s_k \) corresponding to a sample that is dropped \((m_k = 1)\), is not considered a sampling instant in the discrete-time model of [10]. In this perspective, a new (longer) sampling interval is considered to be the difference between the sampling instants of two successive samples that are not dropped. Since the dropped samples are ignored, we introduce a new counter \( l \) and sampling interval \( \tilde{h}_l \). For each \( l \) there exists a corresponding sample \( k_l \) given by
\[
k_{l+1} = \min\{k > k_l \mid m_k = 0\},
\] (3.14)
i.e. \( k_{l+1} = k_l + \delta_{k_{l+1}} + 1 \) for all \( l \in \mathbb{N} \). Defining new states and inputs on these new sample instants \( s_{k_l} \) as \( \tilde{x}_l = x_{k_l} \) and \( \tilde{u}_l = u_{k_l} \), results in the following discrete-time NCS model
\[
\tilde{x}_{l+1} = e^{Ah_l}\tilde{x}_l + \int_{0}^{\tilde{h}_l-\tilde{\tau}_l} e^{As}dB\tilde{u}_l + \int_{\tilde{h}_l-\tilde{\tau}_l}^{\tilde{h}_l} e^{As}dB\tilde{u}_{l-1}.
\] (3.15)

In (3.15), \( \tilde{h}_l \in [\delta+1][h_{\min}, h_{\max}] \) is the prolongated sampling interval. By applying a static state feedback controller \( \tilde{u}_l = -\hat{K}\tilde{x}_l \) to (3.15) and introducing the augmented state vector \( \xi_l := [\tilde{x}_l^T \ \tilde{u}_{l-1}^T]^T \), we obtain
\[
\xi_{l+1} = \tilde{A}_2(\tilde{h}_l, \tilde{\tau}_l)\xi_l,
\] (3.16)
where
\[
\tilde{A}_2(\tilde{h}_l, \tilde{\tau}_l) = \begin{bmatrix}
e^{Ah_l} - \int_{0}^{\tilde{h}_l-\tilde{\tau}_l} e^{As}dB\hat{K} & \int_{\tilde{h}_l-\tilde{\tau}_l}^{\tilde{h}_l} e^{As}dB \\
-\hat{K} & 0
\end{bmatrix}.
\] (3.17)
### 3.2.3 Explicit Dropout Modelling

Let us now present a hybrid discrete-time NCS model that explicitly models the occurrence of packet drops. As shown in (3.3), at each sampling instant \( k \) a packet may drop or not. Exact discretisation of (3.4) at sampling instants \( s_k \) for both cases results in the following discrete-time uncertain NCS model

\[
x_{k+1} = \begin{cases} e^{Ah_k}x_k + \int_{h_k}^{\tau_k} e^{As}dBu_k + \int_0^{h_k-\tau_k} e^{As}dBu_k^\text{old} & \text{if } m_k = 0, \\
e^{Ah_k}x_k + \int_0^{h_k} e^{As}dBu_k^\text{old} & \text{if } m_k = 1,
\end{cases}
\]

(3.18)

with \( u_k := u(s_k) \) the controller output at the \( k \)th sampling instant and \( u_k^\text{old} = u_l \), where \( l = \max\{l \in \mathbb{N}|s_l < s_k \land m_l = 0\} \), the previously computed and successfully implemented controller output.

The model in (3.18) will form the basis for a newly introduced model. We apply a static state feedback control law of the form \( u_k = -\bar{K}x_k \) to the model and define the state of the closed-loop NCS model as \( \xi_k := \begin{bmatrix} x_k^\top & (u_k^\text{old})^\top \end{bmatrix}^\top \), which results in the following discrete-time hybrid system

\[
\xi_{k+1} = \tilde{A}_3(h_k, \tau_k, m_k)\xi_k,
\]

(3.19)

where

\[
\tilde{A}_3(h_k, \tau_k, 0) = \begin{bmatrix} e^{Ah_k} - \int_0^{h_k-\tau_k} e^{As}dBK & \int_0^{h_k-\tau_k} e^{As}dB \\ -\bar{K} & 0 \end{bmatrix},
\]

(3.20a)

\[
\tilde{A}_3(h_k, \tau_k, 1) = \begin{bmatrix} e^{Ah_k} & \int_0^{h_k} e^{As}dB \\ 0 & I \end{bmatrix}.
\]

(3.20b)

Combining this with (3.6) and (3.7), we obtain the complete hybrid model, consisting of two modes in which the dynamics are described by uncertain linear systems, as depicted in Fig. 3.3.

### 3.3 Overapproximation of Model

Direct stability analysis on the models from Section 3.2 is difficult as the infinite number of delays and sampling intervals in \([\tau_{\min}, \tau_{\max}]\) and \([h_{\min}, h_{\max}]\), respectively, and the nonlinear appearance of these uncertain parameters obstruct a finite number of stability conditions. One remedy is to overapproximate systems (3.9), (3.16) and (3.19) by a system in which the uncertainties appear in a polytopic and/or additive manner. This can be achieved by using one of the available overapproximation methods (see, e.g., [4, 9, 11, 12, 8, 23]). Here, we take the method from [4] that is
based on the Jordan form of the continuous-time system matrix $A$, although the other methods can be used as well. We apply this overapproximation method on the hybrid model from (3.19) and (3.20) and provide the corresponding stability conditions. A similar approach is applied to the models from Sections 3.2.1 and 3.2.2 as well, to obtain the numerical results in Section 3.6. For the sake of brevity, we only work out the details for the model presented in Section 3.2.3. However, the results can also be applied to the models of Sections 3.2.1 and 3.2.2 in a similar fashion.

Basically, we express the matrix $A$ as $A = TJT^{-1}$ with $J$ the real Jordan form of $A$ and $T$ an invertible matrix that contains the (generalised) eigenvectors of $A$. With this real Jordan matrix, the exponential term $e^{At} = Te^{Jt}T^{-1}$ has an useful general structure that can be used to obtain a model in which the uncertain parameters $h_k$, $\tau_k$ and $m_k$ appear explicitly, i.e.

$$\xi_{k+1} = \left( F_0(m_k) + \sum_{i=1}^{\zeta} \alpha_i(h_k, \tau_k) F_i(m_k) \right) \xi_k,$$  

(3.21)

where $\zeta$ is the number of time-varying functions $\alpha_i(\cdot)$ given by 2$\nu$ with $\nu \leq n$, where $n$ is the dimension of the state vector $x$. We have $\nu = n$ when each distinct eigenvalue of $A$ corresponds to one Jordan block only and $\nu < n$ otherwise. The functions $\alpha_i(h_k, \tau_k)$ are typically of the form $(h_k - \tau_k)^{j-1}e^{\lambda(h_k - \tau_k)}$, if $\lambda$ is a real eigenvalue of $A$, and $(h_k - \tau_k)^{j-1}e^{\alpha(h_k - \tau_k)\cos(b(h_k - \tau_k))}$ or $(h_k - \tau_k)^{j-1}e^{\alpha(h_k - \tau_k)\sin(b(h_k - \tau_k))}$, when $\lambda$ corresponds to a pair of complex conjugate eigenvalues ($\lambda = a \pm b\sqrt{-1}$) of $A$. Here, $j = 1, 2, \ldots, r$, where $r$ is the size of the largest Jordan block corresponding to $\lambda$.

In particular, for large systems (large $\nu$) this might be necessary to use, as otherwise the LMIs become prohibitively complex for today’s LMI solvers. By using bounds on the uncertain parameters $h_k$ and $h_k - \tau_k$, we obtain the following sets of matrices

$$\mathcal{F}_0 = \left\{ F_0(0) + \sum_{i=1}^{\zeta} \alpha_i(h_k, \tau_k) F_i(0) \mid (h_k, \tau_k) \in \Theta \right\}$$  

(3.22)

and

$$\mathcal{F}_1 = \left\{ F_0(1) + \sum_{i=1}^{\nu} \alpha_i(h_k) F_i(1) \mid h_k \in \Theta \right\},$$  

(3.23)

that contain all possible matrix combinations in (3.21). The set in (3.23) only has $\nu$ uncertainty functions $\alpha_i$ as (3.20b) only depends on $h_k$ and not on $h_k - \tau_k$. Note that $\alpha_i(h_k, \tau_k)$ is arranged as such that $\alpha_i(h_k, \tau_k)$ for $i \in \{1, 2, \ldots, \nu\}$ only depends on $h_k$. To overcome the infinite size of the sets $\mathcal{F}_0$ and $\mathcal{F}_1$, a polytopic overapproximation of the sets is pursued. Denote the maximum and minimum value of $\alpha_i(h_k, \tau_k)$, respectively, by

$$\bar{\alpha}_i = \max_{(h_k, \tau_k) \in \Theta} \alpha_i(h_k, \tau_k), \quad \underline{\alpha}_i = \min_{(h_k, \tau_k) \in \Theta} \alpha_i(h_k, \tau_k),$$  

(3.24)

with $\Theta$ defined in (3.5). Then it is readily seen that the sets of matrices $\mathcal{F}_0$ and $\mathcal{F}_1$, are subsets of the convex hulls $\text{co}\{\mathcal{H}_{\mathcal{F}_0}\}$ and $\text{co}\{\mathcal{H}_{\mathcal{F}_1}\}$, i.e. $\mathcal{F}_0 \subseteq \text{co}\{\mathcal{H}_{\mathcal{F}_0}\}$ and $\mathcal{F}_1 \subseteq \text{co}\{\mathcal{H}_{\mathcal{F}_1}\}$, with

$$\mathcal{H}_{\mathcal{F}_0} = \left\{ F_0(0) + \sum_{i=1}^{\zeta} \alpha_i F_i(0) \mid \alpha_i \in \{\underline{\alpha}_i, \bar{\alpha}_i\}, i = 1, 2, \ldots, \zeta \right\}$$  

(3.25)

and

$$\mathcal{H}_{\mathcal{F}_1} = \left\{ F_0(1) + \sum_{i=1}^{\nu} \alpha_i F_i(1) \mid \alpha_i \in \{\underline{\alpha}_i, \bar{\alpha}_i\}, i = 1, 2, \ldots, \nu \right\}.$$

(3.26)
3.4 Stability Analysis of NCS

Stability analysis can be performed on the overapproximated NCS with the set of vertices in (3.25) and (3.26). For enumeration purposes, we will write the set of vertices \( \mathcal{H}_{\mathcal{F}_0} \) as \( \mathcal{H}_{\mathcal{F}_0} = \{ H_{\mathcal{F}_0, l_0} \mid l_0 = 1, 2, \ldots, 2^\nu \} \) and \( \mathcal{H}_{\mathcal{F}_1} \) as \( \mathcal{H}_{\mathcal{F}_1} = \{ H_{\mathcal{F}_1, l_1} \mid l_1 = 1, 2, \ldots, 2^\nu \} \). Using these finite sets of \( 2^\nu \) and \( 2^\nu \) vertices, respectively, a finite number of LMI conditions to analyse stability is formulated in the following theorem. Note that the LMIs given in this theorem are only applicable for the model of Section 3.2.3. For the LMI conditions of the other two approaches, the reader is referred to [5].

**Theorem 3** Consider the NCS model (3.4) and its discrete-time representation (3.19) with (3.20) for sequences of sampling instants, delays and packet dropouts \( \{ h_k, \tau_k, m_k \}_{k \in \mathbb{N}} \), where \( (h_k, \tau_k) \in \Theta \), as in (3.5), and \( m_k \) satisfying (3.6). Consider the equivalent representation (3.21) based on the real Jordan form of \( A \) and the set of vertices \( \mathcal{H}_{\mathcal{F}_0} \), as in (3.25), and \( \mathcal{H}_{\mathcal{F}_1} \), as in (3.26).

If there exist a set of symmetric positive definite matrices \( \{ P_0, \ldots, P_3 \} \) and a scalar \( \gamma \in [0, 1) \) satisfying

\[
\begin{bmatrix}
(1 - \gamma)P_i & H_{\mathcal{F}_{0, l_0}}^T P_0

P_0 H_{\mathcal{F}_{0, l_0}} & P_0
\end{bmatrix} \succ 0
\]

and

\[
\begin{bmatrix}
(1 - \gamma)P_j & H_{\mathcal{F}_{1, l_1}}^T P_{j+1}

P_{j+1} H_{\mathcal{F}_{1, l_1}} & P_{j+1}
\end{bmatrix} \succ 0,
\]

for all \( i \in \{0, \ldots, 3\} \), \( j \in \{0, \ldots, 3\} \), \( l_0 \in \{1, 2, \ldots, 2^\nu\} \) and \( l_1 \in \{1, 2, \ldots, 2^\nu\} \), then the closed-loop NCS from (3.4) is globally asymptotically stable.

**Proof:** The proof is based on showing that \( V(\xi_k, \delta_k) = \xi_k^T P_{\delta_k} \xi_k, \delta_k \in [0, \bar{\delta}] \), is a parameter-dependent LF for the system in (3.21). First of all, due to the finite number of matrices \( P_i \) and each \( P_i \) is positive definite, there exist \( 0 < c_1 \leq c_2 \) such that \( c_1 \|\xi_k\|^2 \leq V(\xi_k, \delta_k) \leq c_2 \|\xi_k\|^2 \) for all \( \xi_k \) and \( \delta_k \). Global asymptotic stability of \( \xi_k = 0 \) is now implied by requiring that \( V \) is strictly decreasing in the sense that

\[
V(\xi_{k+1}, \delta_{k+1}) - V(\xi_k, \delta_k) = \xi_{k+1}^T P_{\delta_{k+1}} \xi_{k+1} - \xi_k^T P_{\delta_k} \xi_k < -\gamma \xi_k^T P_{\delta_k} \xi_k,
\]

where \( (\xi_{k+1}, \delta_{k+1}) \) and \( (\xi_k, \delta_k) \) are related through (3.19) and (3.7), subject to (3.6). Since, depending on the transition between the two modes of Fig. 3.3, \( \delta_{k+1} \) can be either \( \delta_{k+1} = 0 \) (i.e. \( m_k = 0 \)) or \( \delta_{k+1} = \delta_k + 1 \) (i.e. \( m_k = 1 \)), we have that (3.29) is satisfied if

\[
\begin{cases}
(1 - \gamma)P_i - A_3^T(h_k, \tau_k, 0) P_0 A_3(h_k, \tau_k, 0) > 0 \\
(1 - \gamma)P_j - A_3^T(h_k, \tau_k, 1) P_{j+1} A_3(h_k, \tau_k, 1) > 0
\end{cases}
\]

for all \( i \in \{0, \ldots, 3\} \) and \( j \in \{0, \ldots, 3\} \). Now note that \( A_3(h_k, \tau_k, 0) \in \mathcal{F}_0 \subseteq \text{co} \{ \mathcal{H}_{\mathcal{F}_0} \} \) and \( A_3(h_k, \tau_k, 1) \in \mathcal{F}_1 \subseteq \text{co} \{ \mathcal{H}_{\mathcal{F}_1} \} \) for all \( (h_k, \tau_k) \in \Theta \). It therefore suffices to prove (3.30) for the vertices of the sets \( \text{co} \{ \mathcal{H}_{\mathcal{F}_0} \} \) and \( \text{co} \{ \mathcal{H}_{\mathcal{F}_1} \} \), as in (3.25) and (3.26), resulting in the following finite set of LMIs

\[
\begin{cases}
(1 - \gamma)P_i - H_{\mathcal{F}_{0, l_0}}^T P_0 H_{\mathcal{F}_{0, l_0}} > 0 \\
(1 - \gamma)P_j - H_{\mathcal{F}_{1, l_1}}^T P_{j+1} H_{\mathcal{F}_{1, l_1}} > 0
\end{cases}
\]

for all \( i \in \{0, \ldots, 3\} \), \( j \in \{0, \ldots, 3\} \), \( l_0 \in \{1, 2, \ldots, 2^\nu\} \) and \( l_1 \in \{1, 2, \ldots, 2^\nu\} \). Applying a Schur complement results in (3.27) and (3.28). Hence, (3.27)-(3.28) imply (3.29), which proves global asymptotic stability of (3.19).

3.5 Discussion

In this section, the advantages and drawbacks of the existing approaches are discussed and we explain why we proposed the presented extensions of the existing methods.
In Section 3.2.1, dropouts are modelled as prolongations of the delays based on [4] and [3]. In [3] a model is given for large delays incorporating dropouts, whereas in Section 3.2.1 a model is presented that includes small delays and dropouts. Due to the general case considered in [3] (large delays), it is possible that \( \delta + 2 \) different control values are active in the interval \([s_k, s_{k+1}]\), while in case of small delays this can be at most two. As a consequence, at most two \( M_j \)'s are nonzero in the model in (3.9) and (3.10) from Section 3.2.1. Here, we use the general result as presented in [4, 3] in which more than two \( M_j \)'s can be nonzero, although it can be easily remedied of course. Moreover, in [3] the dropout counter \( \delta_k \) is not used in the model. To be precise, the relation between \( \delta_{k+1} \) and \( \delta_k \) as in (3.7) is not exploited to reduce conservatism. Therefore, the number of successive dropouts at the next sampling instant \( s_{k+1} \) is not related to the number of dropouts at the current instant \( s_k \). This may lead to spurious effects, such as for example \( \delta_{k+1} = \delta_k = 1 \) is possible, which means that a particular control input is considered to be dropped in one sampling interval while it is considered not to be dropped in the next. Clearly, this is not possible in reality. If we would resolve these two issues, which is rather straightforward for the case of small delays, the result becomes actually of a similar nature as the method of Section 3.2.3. For comparison reasons we kept here the general method of [4, 3] that also applies to the large delay case in which such simplifications are not so obvious or even impossible.

In Section 3.2.2, dropouts are modelled as prolongations of the sampling intervals. Regarding the work in [10], it is unclear which overapproximation method is used to arrive at a polytopic embedding of the system as in (3.25) and (3.26). To implement this approach, we have proposed the real Jordan overapproximation technique discussed in Section 3.3. Secondly, García-Rivera and Barreiro [10] use a NCS with constant sampling intervals. We extended this to a discrete-time NCS model with varying sampling intervals, as shown in (3.15). The transformation of the range of sampling intervals can be taken as \( h_k \in H_1 := [\min, (\delta + 1)\max] \), which contains spurious values for \( h_k \) when \( \min > 0 \), because all intervals are merged into one large interval instead of considering them as separate intervals. We reduced this source of conservatism by defining the range as \( h_1 \in H_u := \bigcup_{\delta=0}^{\delta}(\delta + 1)[\min, \max] \) such that no spurious sampling intervals occur when \( \min > 0 \). Considering \( H_u \) requires to perform the overapproximation for each individual (non-overlapping) interval and one can assign one Lyapunov function (LF) to each interval such that a parameter-dependent LF \( V(\tilde{x}_{t+1}, \delta) \) is obtained as in the proof of Theorem 3. In case subsequent intervals \( \delta'[\min, \max] \) and \( (\delta + 1)[\min, \max] \) overlap, one can actually unite them in one new interval \( \delta'[\min, \max], (\delta + 1)[\min, \max] \) and use only one LF for this united interval, which reduces the number of LMIs (but increases conservatism). Of course, one can also consider one common LF \( V(\tilde{x}_t) \). With a common \( P \) and taking one large interval as in \( H_1 \), the least number of LMIs is obtained, but the conservatism is increased. Note that in case of \( \min = 0 \), the interval ranges \( H_1 \) and \( H_u \) are equal.

A general remark is that both [3] and [10] use a common quadratic Lyapunov function (CQLF) to perform stability analysis on the models. As shown in Section 3.4, we propose to use a parameter-dependent Lyapunov function (PDLF) that depends on the dropout counter \( \delta_k \). Below we will apply this PDLF for an example. Note that \( \delta_k \) is not used in the model based on [3] and thus no \( \delta_k \)-dependent LF can be applied to that approach. As already mentioned, although PDLFs reduce the conservatism of the stability of NCSs\(^2\), they also lead to more LMIs than CQLFs. This higher number of LMIs results in a more complex stability analysis. The number of LMIs for all modelling methods are given in Table 3.1 for CQ and PD LFs. Note that, the explicit dropout modelling approach (Section 3.2.3) is not feasible for CQLFs, because the second mode of the automaton is always unstable.

---

\(^2\)One can reduce the conservatism of NCSs further by using PDLFs that also depend on the delay and sampling (or actually on the vertices of the polytopic overapproximation in (3.25) and (3.26)) at the cost of more LMIs. For ease of exposition, we use dependency on \( \delta_k \) only.
Table 3.1: Number of LMIs for the different modelling methods for varying sampling intervals and delays.

<table>
<thead>
<tr>
<th></th>
<th>CQLF</th>
<th>PDLF</th>
</tr>
</thead>
<tbody>
<tr>
<td>[3] small delays</td>
<td>$2^{(\delta+2)\nu}$</td>
<td>X</td>
</tr>
<tr>
<td>[10] with $H_l$</td>
<td>$2^{2\nu}$</td>
<td>X</td>
</tr>
<tr>
<td>[10] with $H_u$</td>
<td>$2^{2\nu}(\delta + 1)$</td>
<td>$2^{2\nu}(\delta + 1)^2$</td>
</tr>
<tr>
<td>Explicit model</td>
<td>X</td>
<td>$2^{2\nu}(\delta + 1) + 2^\nu \delta$</td>
</tr>
</tbody>
</table>

3.6 Illustrative Example

In this section, we show the results of the stability analysis for the double integrator example, given by $\dot{x} = Ax + Bu$, where

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$ (3.32)

Here, we limit ourselves to a fixed sampling interval $h = h_{\text{min}} = h_{\text{max}} = 1\text{ms}$, where a different number of subsequent packet dropouts $\bar{\delta}$ can occur, in combination with a time-varying delay that is upper bounded by $\tau_{\text{max}}$, which is equal or smaller than the fixed sampling interval ($\tau_{\text{min}} = 0, \tau_{\text{min}} \leq \tau_{\text{max}} \leq h$). The gain of the static state feedback controller $u_k = -\hat{K}x_k$ is given by $\hat{K} = \begin{bmatrix} 6000 & K_b \end{bmatrix}$. We are interested in the stability region, i.e. all values of $K_b$ that guarantee stability for each combination of delays, satisfying $\tau_k \in [0, \tau_{\text{max}}]$ and sequences of packet dropouts, satisfying (3.6). Based on the conditions in [5] and the ones in Theorem 3, we obtain the stability regions in Fig. 3.4 and Fig. 3.5, given by an upper bound for $K_b$, for $\bar{\delta} = 0$, i.e. no packet dropouts, $\bar{\delta} = 1$ and $\bar{\delta} = 2$. Note that $K_b$ also has lower bounds, which are not shown in Fig. 3.4 and Fig. 3.5. The reason for this is that these are small ($K_b < 20$ for all approaches and for all values of $\tau_{\text{max}}$ and $\delta$) in comparison with the upper bounds on $K_b$.

Clearly, packet dropouts decrease the allowable controller gains that stabilise the NCS. To illustrate that the presented explicit dropout modelling approach reduces conservatism with respect to the approach presented in [3], their results are also shown in Fig. 3.4. Note that, in the approach of [3] the maximal controller gain $K_b$ for which stability is guaranteed, for a maximal delay equal to the sampling interval ($\tau_{\text{max}} = h$) without dropouts ($\delta = 0$), is the same as the gain for one dropout with no delay ($\delta = 1, \tau_{\text{max}} = 0$), denoted by the dash-dotted line in Fig. 3.4. The same holds for more dropouts, e.g. $K_b$ corresponding to ($\tau_{\text{max}} = h, \delta = 1$) for which stability can be proven, equals $K_b$ corresponding to ($\tau_{\text{max}} = 0, \delta = 2$). This is a consequence of the modelling technique as discussed in Section 3.2.1, i.e. a dropout without delay is modelled as a delay that is equal to the sampling interval $\tau_{\text{max}} = h$. Essentially, this is caused by the computation of the bounds $t_{j,\text{min}}$ and $t_{j,\text{max}}$ or the control update instants.

Fig. 3.4 also shows that the approach from [10] (extended with $H_u$) gives similar stability regions as the newly proposed explicit modelling approach, i.e. for some $\tau_{\text{max}}$ the extension of [10] is less conservative and for some $\tau_{\text{max}}$ the newly proposed explicit modelling method is less conservative. Note that, as mentioned in Section 3.5, Garcia-Rivera and Barreiro use a common quadratic LF and the newly proposed method uses a $\delta_k$-dependent LF. Introducing a parameter-dependent LF to the extended approach from [10], results in slightly less conservative stability regions than the CQLF results. Though, the results are still comparable with the results of the newly proposed method, as shown in Fig. 3.5. This observation turns out to be true for various examples. As such, from a conservatism point of view, the newly proposed method and the extended method based on [10] seem to be comparable. Note though that the numerical complexity figures (number of LMIs) are different for both methods (see Table 3.1).
Figure 3.4: Upper bounds of the stability region for \( \bar{K} = [6000 \ K_\delta] \) for different values of \( \bar{\delta} \) and 
\( \tau \in [0, \tau_{\text{max}}] \).

Figure 3.5: Upper bounds of the stability region for \( \bar{K} = [6000 \ K_\delta] \) for different values of \( \bar{\delta} \) and 
\( \tau \in [0, \tau_{\text{max}}] \).
3.7 Conclusions

In this paper, we studied the stability of networked control systems (NCSs) using a discrete-time model that incorporates time-varying sampling intervals, time-varying delays and packet dropouts. Existing approaches modelled packet dropouts as prolongations of the delay or prolongations of the sampling interval. Here, we proposed a third method based on modelling packet dropouts explicitly leading to a discrete-time hybrid automaton-based NCS model. By doing so, the phenomenon of packet dropouts is truly modelled and no spurious effects are introduced as in the case of modelling the dropouts as prolongation of the delays or sampling intervals. Based on this new model, in which hybrid modelling plays an important role, constructive LMI conditions for analysing stability of the closed-loop system are derived. In addition, we extended the method from [10] that is based on prolongation of the sampling interval in three ways, namely (i) the range of sampling intervals is redefined, (ii) the model is overapproximated using the real Jordan form and (iii) a parameter-dependent Lyapunov function is introduced.

After reviewing the three modelling methods based on their perspectives, we compared them numerically using the double integrator example. We showed that the newly proposed approach reduces conservatism when compared to the delay prolongation approach from [3]. We also showed that the newly proposed method, in combination with a parameter-dependent LF, gives similar stability regions as the extended sampling interval prolongation technique from [10], in combination with a common quadratic or parameter-dependent LF.
Chapter 4

The Matlab Toolbox

As mentioned in the introduction, a Matlab toolbox will make the available analysis techniques of NCSs more accessible for both academia and industry. A major part of this master’s project consisted of creating such a toolbox. In this chapter, the developed Matlab toolbox is discussed in detail and it is shown how this toolbox can be used to analyse stability of NCSs. As mentioned in Chapters 2 and 3, many different (dropout) modelling, overapproximation and analysis approaches are proposed in the literature (see, e.g., [4, 10, 8, 9, 23, 12, 11, 5] and the references therein). In this chapter, we focus on two of the ways to model NCSs with dropouts, as discussed in the previous chapter, and on the two methods to overapproximate the resulting models, as discussed in Chapter 2. Tools to study NCS models are programmed in Matlab, i.e.

- Modelling NCSs with dropouts:
  - explicit dropout modelling using hybrid automata and
  - modelling dropouts as prolongation of the sampling interval.

- Overapproximation of the models:
  - based on the real Jordan form (see, e.g., [22]) and
  - based on the Cayley-Hamilton theorem (see, e.g., [11]).

The first method that is discussed in Section 3.2 is not included in the toolbox, because the results of this method are more conservative than the other modelling approaches (see Section 3.6).

In this chapter, the structure and components of the Matlab toolbox are discussed and clarified. In Fig. 4.1, the global structure of the toolbox is depicted in a schematic manner. Note that the global structure of the toolbox is the same as the structure of Chapter 2 and 3, i.e.

- model the NCS using one of the available dropout modelling approaches,
- overapproximate the resulting discrete-time NCS model by a polytopic model and
- perform stability analysis using Lyapunov-based stability conditions.

This chapter has a similar structure. Here, all Matlab files that are part of the Matlab toolbox are given in the list on page 27 and they will be referred to using a bold typeface. All these m-files have a Matlab documentation, which is accessible by typing `help <name of the m-file>` in the Matlab command window, in which the working principle of the routine and how to use the routine, are briefly explained.
Dropout modelling as prolongation of the sampling interval

Controller synthesis

Controller \( K \) available?

Stability analysis using Lyapunov functions

Common quadratic Lyapunov function

Parameter-dependent Lyapunov function

The NCS is not stabilisable using state feedback and the chosen type of LF

The NCS is stable

No Controller \( K \) available? Yes

Apply control law \( u_k = -Kx_k \) to the overapproximated NCS model

Stabilizable controller?

No

Yes

The NCS is stable or no conclusions about stability can be drawn

Inputs: \( \{A, B, K, [h_{\min}, h_{\max}], \tau_{\min}, \tau_{\max}, \delta, \epsilon, \gamma, \text{ type of Lyapunov function} \} \)

Figure 4.1: Global structure of the toolbox.
The Matlab toolbox can be used by running one of the following six main files that will be discussed in this chapter. Dropouts are modelled explicitly using automata in the routines

- Automaton_JNF.m
- Automaton_CH1.m
- Automaton_CH2.m

and dropouts are modelled as prolongation of the sampling intervals in the routines

- Garcia_JNF.m
- Garcia_CH1.m
- Garcia_CH2.m

These main files make use of the following functions, which will also be discussed briefly in this chapter.

- JordanEig.m
- JordanBlocks.m
- Jsymbolic.m
- exponent.m
- exponent_complex.m
- integral_ht_h.m
- integral_lambda0_ht_h.m
- integral_complex_ht_h.m
- integral_0_ht.m
- Gamma_Gamma0.m
- Cayley.m
- check_alpha.m
- constant_sampling.m
- varying_sampling.m
- CQLF.m
- PDLF.m
- LMI.m
- LMI_common.m
- LMI_dependent.m

An important note is that one needs the Linear Systems Toolkit [17] to make use of the Matlab toolbox described here to determine the real Jordan form.
4.1 Matlab Implementation Without Dropouts

To simplify the explanation of the Matlab toolbox, we consider the NCS that only incorporates time-varying sampling intervals and time-varying communication delays. The case that dropouts are present can be handled analogously in the toolbox and is discussed in Section 4.2. The general NCS model, with the lifted state vector $\xi_k$, is given by (2.3), i.e.

$$
\xi_{k+1} = \begin{bmatrix}
e^{Ah_k} & \int_{h_k}^{h_k+\tau_k} e^{As} ds B \\
0 & 0
\end{bmatrix}
\begin{bmatrix}
\xi_k \\
0
\end{bmatrix}
+ \begin{bmatrix}
\int_0^{h_k-\tau_k} e^{As} ds B \\
I
\end{bmatrix}
\begin{bmatrix}
u_k
\end{bmatrix}
+ \begin{bmatrix}
\int_0^{h_k-\tau_k} e^{As} ds B \\
I
\end{bmatrix} u_k.
$$

(4.1)

4.1.1 Overapproximation in the Matlab Toolbox

As discussed in Section 2.1, both the real Jordan form based and Cayley–Hamilton theorem based overapproximation techniques use in principle the real Jordan form. Hence, we will first discuss the derivation of the real Jordan form and determine its structural properties.

The first step in the overapproximation process is to determine all eigenvalues to decide what classes of eigenvalues are present in the system: non-zero and real, zero or complex [eigSym lambda] = JordanEig(J), where eigSym contains all symbolic eigenvalues, lambda contains all real eigenvalues and $J$ is the real Jordan matrix. Next, all Jordan blocks are separated in $JJ$ = JordanBlocks(J) (where $JJ$ is a cell that contains all separated Jordan blocks) by checking for ones on the upper diagonal (in case of a real eigenvalue) or non-zero elements on the lower diagonal and ones on the second upper diagonal (in case of a complex eigenvalue pair).

Because Matlab performs better with symbolic expressions, the entire real Jordan matrix $J$ and all Jordan blocks $J_h$ are transformed to symbolic matrices in the file $Jsym = Jsymbolic(J,eigSym)$ (where $Jsym$ is the real Jordan matrix with symbolic eigenvalues), such that all nonzero eigenvalues are symbols instead of (rounded) numbers. An eigenvalue that is smaller than $10^{-5}$ is assumed to be zero and is not replaced by a symbol.

Now that all Jordan blocks are separated, the exponential and integral terms in (4.1) can be determined symbolically by Matlab, using the expressions

$$
e^{Ah_k} = Q \text{diag}(e^{A_1 h_k}, e^{A_2 h_k}, \ldots, e^{A_p h_k}) Q^{-1}
$$

(4.2)

and

$$
\int e^{As} ds = Q \text{diag} \left( \int e^{J_1 s} ds, \int e^{J_2 s} ds, \ldots, \int e^{J_p s} ds \right) Q^{-1}
$$

(4.3)

from Chapter 2 in combination with (2.11) and (2.12). With this, the matrix exponential $e^{J_h k}$ can be written as the sum of varying functions $\alpha_j$ and constant matrices $S_j$ for all $i = 1, 2, \ldots, p$, i.e.

$$
e^{J_h k} = \sum_{j=1}^{\nu} \alpha_{1,j}(h_k) S_{1,j},
$$

(4.4)

with $\nu \leq n$ as discussed in Chapter 2. In case $J_h$ corresponds to a real eigenvalue $\text{lambda}$, $\alpha_{1,j}(h_k)$ and $S_{1,j}$ are determined in the file [S1 alfa1] = exponent(JJ{i},eigSym,lambda), where eigSym is the symbolic eigenvalue, and in case $J_h$ corresponds to a complex eigenvalue pair, $\alpha_{1,j}(h_k)$ and $S_{1,j}$ are determined in the file [S1 alfa1 Jc sigma omega] = exponent_complex(JJ{i}), where $S1$ is a cell that contains all matrices $S_{1,j}$, $alfa1$ contains all varying functions $\alpha_{1,j}(h_k)$ and $Jc$ is the real Jordan block for a complex eigenvalue pair $\lambda_i = \sigma + \omega \sqrt{-1}$.

The same approach is used to express the exponential terms of the NCS model (4.1) symbolically, i.e.

$$
\int_{h_k-\tau_k}^{h_k} e^{J_h s} ds = \sum_{i=1}^{2\nu} \alpha_{2,i}(h_k, \tau_k) S_{2,i},
$$

(4.5)

where $\alpha_{2,i}(h_k, \tau_k)$ (alfa2) and $S_{2,i}$ (S2) are calculated in
\[ [S2 \text{ alfa2}] = \text{integral}_\text{-ht}_\text{-h}(\text{JsymSep}\{i\}, \text{eigSym}, \lambda) \] (if \( J_i \) corresponds to a real and nonzero eigenvalue),

\[ [S2 \text{ alfa2}] = \text{integral}_\text{-lambda0}_\text{-ht}_\text{-h}(\text{JJ}\{i\}) \] (if \( J_i \) corresponds to an eigenvalue that equals zero) or

\[ [S2 \text{ alfa2}] = \text{integral}_\text{-complex}_\text{-ht}_\text{-h}(\text{Jc}, \text{sigma}, \text{omega}) \] (if \( J_i \) corresponds to a complex eigenvalue pair).

Here, \( \text{JsymSep} \) is a cell that contains the symbolic versions of the separated Jordan blocks. The second integral term is given by

\[
\int_0^{h_k-\tau_k} e^{Js} ds = S_{3,0} + \sum_{j=1}^{\nu} \alpha_{3,j}(h_k, \tau_k)S_{3,j}, \tag{4.6}
\]

which is determined symbolically by using the results from (4.5), because the lower integration bound of (4.5) is equal to the upper integration bound of (4.6), i.e. the functions \( \alpha_{3,j} \) are already contained in \( \alpha_{2,l} \) and the constant matrices \( S_{3,j} \) are contained in \( S_{2,l} \), for \( j = 1, 2, \ldots, \nu \) and \( l = 1, 2, \ldots, 2\nu \). The resulting \( \alpha_{3,j} \), \( S_{3,0} \) and \( S_{3,j} \) are determined in \([G0 S3 alfa3] = \text{integral}_0 \text{-ht}(\text{alfa22}, \text{S22}, \text{Bc}, \text{Q})\), independent of the type of eigenvalue that corresponds to \( J_i \). In this routine, \( G0 \) is equal to \( S_{3,0} \). \( S3 \) is a cell that contains all matrices \( S_{3,j} \), \( alfa3 \) contains all varying functions \( \alpha_{3,j} \) and \( alfa22 \) and \( S22 \) contain all functions \( \alpha_{2,l} \) and all matrices \( S_{2,l} \) from eq. (4.5).

From this point, the Cayley-Hamilton and real Jordan based overapproximation methods take different routes, because the Cayley-Hamilton theorem based method uses different varying functions and constant matrices than the approach based on the real Jordan form.

### 4.1.2 Overapproximation Using the Real Jordan Form

Now that each term of the NCS model (4.1) is written as the sum of varying functions \( \alpha_{i,j} \) and constant matrices \( S_{i,j} \), it is possible to write the model as in (2.5) by substituting (4.4), (4.5) and (4.6) into (4.1), resulting in

\[
\xi_{k+1} = \left( F_0 + \sum_{i=1}^{2\nu} \alpha_i(h_k, \tau_k)F_i \right) \xi_k + \left( G_0 + \sum_{i=1}^{2\nu} \alpha_i(h_k, \tau_k)G_i \right) u_k, \tag{4.7}
\]

with \( \{\alpha_1, \ldots, \alpha_{2\nu}\} \) is the total set of varying functions that contains all different functions from \( \alpha_{1,j} \), \( \alpha_{2,l} \) and \( \alpha_{3,j} \), where the objective is that each distinct varying function only appears once in \( \{\alpha_1, \ldots, \alpha_{2\nu}\} \). This is done in the routine \([\text{alfa new}, \text{Ah}, \text{Bh}] = \text{check_alpha}(\text{alfa11}, \text{alfa22}, \text{alfa33}, \text{S11}, \text{S22}, \text{S33}, \text{Q}, \text{Bc}, \text{J}, \text{y})\) that puts all functions from \( \alpha_{1,j} \) (\( \text{alfa11} \)), \( \alpha_{2,l} \) (\( \text{alfa22} \)) and \( \alpha_{3,j} \) (\( \text{alfa33} \)) in one set \( \alpha_i \) (\( \text{alfa} \_\text{new} \)) and then checks each function on multiplicity by using a symbolic subtraction. \text{check}_\text{alpha.m} \) uses the same trick to obtain \( F_0 \) and \( F_i \) (\( \text{Ah} \)), and \( G_0 \) and \( G_i \) (\( \text{Bh} \)) by using the sets of matrices \( S_{1,j} \) (\( \text{S11} \)), \( S_{2,l} \) (\( \text{S22} \)) and \( S_{3,j} \) (\( \text{S33} \)) and system matrix \( B \) (\( \text{Bc} \)). Note that the input \( y \) of \text{check}_\text{alpha.m} \) equals 0 in case the overapproximation method based on the real Jordan form is used.

Since the uncertainties \( h_k \) and \( \tau_k \) can take infinitely many values since \( (h_k, \tau_k) \in \Theta \), for all \( k \in \mathbb{N} \), where \( \Theta \) is as in (2.2), the model in (4.7) needs to be embedded in a convex hull having a finite number of vertices (see (2.17)), i.e.

\[
\mathcal{H}_S = \left\{ \left( F_0 + \sum_{i=1}^{2\nu} \alpha_i F_i \right), \left( G_0 + \sum_{i=1}^{2\nu} \alpha_i G_i \right) \mid \alpha_i \in \{\alpha_{i}, \overline{\alpha}_i\}, \ i = 1, 2, \ldots, 2\nu \right\}, \tag{4.8}
\]

with \( F_0, F_i, G_0 \) and \( G_i \) are constant matrices and \( \alpha_i \) are time-varying functions with minimal values \( \alpha_i \) and maximal values \( \overline{\alpha}_i \) for \( i = 1, 2, \ldots, 2\nu \).
To obtain this set of matrices $\mathcal{H}_S$, all minima and maxima of $\alpha_i$, $\beta_j$, and $\pi_l$, respectively, need to be determined in the Matlab toolbox. This is done in $[A_k,B_k,d_k] = \text{constant\_sampling}(\text{alfa\_new},A_k,B_k,h,\tau,\nu,mk,\text{size}J)$ (in case of constant sampling intervals $h_{\text{min}} = h_{\text{max}} = h$, which results in less varying functions, i.e. $\nu$ instead of $2\nu$) or in $[A_k,B_k,d_k] = \text{varying\_sampling}(\text{alfa\_new},A_h,B_h,h,\tau,\nu,mk,\text{size}J)$ (in case of varying sampling intervals $h_k \in [h_{\text{min}}, h_{\text{max}}]$, where $h_{\text{min}} < h_{\text{max}}$). In these files, all combinations of the minimal and maximal numerical values of all functions $\alpha_i$ are captured in $dk$ with the corresponding constant matrices in the cells $A_k$ and $B_k$. Moreover, the sampling intervals lie in $h = [h_{\text{min}}, h_{\text{max}}]$, the delays lie in $\tau = [\tau_{\text{min}}, \tau_{\text{max}}]$, size$J$ equals the dimension of the state vector $x$ and $mk$ equals the number of dropouts (in this case $mk = 0$). Note that the models from Sections 3.2.2 (3.17) and 3.2.3 (3.20) of the previous chapter are functions of the variable $mk$.

This step concludes the overapproximation process: all vertices $\mathcal{H}_{F,j}$ and $\mathcal{H}_{G,j}$ of the convex embedding are contained in the finite set $\mathcal{H}_S$.

### 4.1.3 Overapproximation Using the Cayley-Hamilton Theorem

As discussed in Section 2.1.2, the overapproximation technique that is based on the Cayley-Hamilton theorem uses summations of different varying functions and constant matrices than the real Jordan based technique, i.e. the constant matrices are powers of the system matrix $A$. In the Matlab toolbox, the varying functions that belong to the constant matrices $A^i$ are determined by using the varying function $\alpha_{1,i}$, $\alpha_{2,i}$ and $\alpha_{3,i}$ that result from the real Jordan form approach, such that the NCS model (4.1) can be written as in (2.36), i.e.

$$
\begin{align*}
\xi_{k+1} = \left[\sum_{j=1}^{\nu} f_{1,j}(h_k) A^{j-1} \sum_{j=1}^{\nu} f_{2,j}(h_k,\tau_k) A^{j-1} B \right] \xi_k + \left[ S_{3,0} + \sum_{j=1}^{\nu} f_{3,j}(h_k,\tau_k) A^{j-1} B \right] u_k,
\end{align*}
$$

(4.9)

In this model, the varying functions $f_{1,j}$ ($\text{alfa1\_new}$), $f_{2,j}$ ($\text{alfa2\_new}$) and $f_{3,j}$ ($\text{alfa3\_new}$) are determined in $[\text{alfa1\_new}, J_i, \text{beta1}] = \text{Cayley}(J_i,S11,\text{alfa11})$, $[\text{alfa2\_new}, J_i, \text{beta2}] = \text{Cayley}(J_i,S22,\text{alfa22})$ and $[\text{alfa3\_new}, J_i, \text{beta3}] = \text{Cayley}(J_i,S33,\text{alfa33})$, respectively, by using

$$
\begin{align*}
f_{1,j} = \sum_{l=1}^{2\nu} \beta_{1,j,l} \alpha_{1,l}, \quad f_{2,j} = \sum_{l=1}^{2\nu} \beta_{2,j,l} \alpha_{2,l} \quad \text{and} \quad f_{3,j} = \sum_{l=1}^{2\nu} \beta_{3,j,l} \alpha_{3,l}
\end{align*}
$$

(4.10)

for all $j = 1,\ldots,\nu$, where $\alpha_{1,l}$, $\alpha_{2,l}$ and $\alpha_{3,l}$ are given in as $\text{alfa11}$, $\text{alfa22}$ or $\text{alfa33}$, respectively. In (4.10) and $\text{Cayley.m}$, the constants $\beta_{i,j,l}$ ($\text{beta1}$, $\text{beta2}$ and $\text{beta3}$) are calculated using the constant matrices $S_{i,l}$ ($S11$, $S22$ and $S33$), which result from the real Jordan form approach, i.e.

$$
\sum_{j=1}^{\nu} \beta_{i,j,l} J^{j-1} = S_{i,l},
$$

(4.11)

for all $l = 1,2,\ldots,\nu$ if $i = 1,3$ and for all $l = 1,2,\ldots,2\nu$ if $i = 2$, where all power of $J$ are contained in the cell $J_i$.

Again, all varying functions that are contained in $f_{1,j}$, $f_{2,j}$ and $f_{3,j}$ are put into one set and each function is checked on multiplicity in $\text{check\_alpha.m}$, so that each varying function only appears once in the total set $\alpha_j$ (see (2.38)). Finally, model (4.9) needs to be overapproximated by a polytopic embedding, i.e.

$$
\mathcal{H}_S = \left\{ \left( F_0 + \sum_{j=1}^{2\nu} \alpha_j F_j \right), \left( G_0 + \sum_{j=1}^{2\nu} \alpha_j G_j \right) \mid \alpha_j \in \{ \alpha_j, \overline{\alpha}_j \}, \; j = 1,2,\ldots,2\nu \right\}.
$$

(4.12)
where $\alpha_j$ and $\bar{\pi}_j$ are determined in `constant_sampling.m` (in case of constant sampling intervals) or in `varying_sampling.m` (in case of varying sampling intervals). This set of matrices $\mathcal{H}_S$ is finite and contains all vertices of the convex overapproximation of the NCS model (4.1).

4.1.4 Apply Available Controller to the Overapproximated Model

If a state feedback $K$ is available, such that the control law $u_k = -Kx_k$, where $x_k$ is the state vector, and the lifted control law $u_k = -\begin{bmatrix} K & 0 \end{bmatrix} \xi_k$, where $\xi_k$ is the lifted state vector, the complete overapproximated closed-loop system can be determined. This is realised by using the overapproximated model (4.8) or (4.12), i.e. the finite sets of matrices $H_{F,j}$ and $H_{G,j}$. Applying the controller to the model, results in the closed-loop overapproximated set given by

$$H_{CL,j} = H_{F,j} - \begin{bmatrix} K & 0 \end{bmatrix} H_{G,j},$$

(4.13)

for all $j = 1, 2, \ldots, 2^{2\nu}$, with $\nu \leq n$ as discussed before.

4.1.5 Stability Analysis in the Matlab Toolbox

With the set of vertices (4.13), the stability of the overapproximated closed-loop NCS model can be analysed. In Section 2.2, we discussed how this is done exactly for common quadratic LFs and parameter-dependent LFs. Because dropouts are not considered, no dropout-dependent LMIs can be constructed and the following set of LMIs is obtained

$$P > \epsilon I,$$

$$H_{CL,j}^T P H_{CL,j} - (1 - \gamma) P < 0,$$

(4.14)

where $0 < \epsilon \leq 1$ is a small scalar that avoids numerical problems in Matlab, $0 \leq \gamma < 1$ ensures the Lyapunov function to decrease with rate $1 - \gamma$ and $H_{CL,j}$ is the set of vertices (4.13) for all $j = 1, 2, \ldots, 2^{2\nu}$. Note that, in this thesis we only considered dropout-dependent LFs and we did not consider dependencies of delay and sampling interval. The LMIs from (4.14) are implemented in the Matlab toolbox in the file `[P, LMIs] = LMI_common(Dt,epsilon,gamma)` (where $P$ is the Lyapunov matrix, the cell `LMIs` contains all solved LMIs and the cell `Dt` contains all vertices of the overapproximation), solved by the SeDuMi (Yalmip) LMI solver [14, 24] and verified if they satisfy the matrix inequality constraints. If they do, the overapproximated closed-loop NCS model (and consequently the original NCS) is stable; if they do not, no conclusions can be drawn about the stability of the NCS.

4.1.6 Controller Synthesis

In case that no controller matrix $K$ is available, the Matlab toolbox has a state feedback synthesis function, based on the theory given in Section 2.3, such that a state feedback of the form $u_k = -Kx_k$ can be obtained. One can choose for a controller $K$ that is synthesised using a common quadratic Lyapunov function ($K = CQLF(A,B,\gamma)$, where $A$ is a cell containing matrices $H_{F,j}$, $B$ is a cell containing matrices $H_{G,j}$ and $\gamma$ is the decay rate of the Lyapunov function) or a parameter-dependent Lyapunov function ($K = PDLF(A,B,\gamma)$). For more information about the controller synthesis technique, the reader is referred to Theorem 2 and [3]. From Theorem 2, the controller synthesis that is based on a parameter-dependent LF (2.58) is implemented in the Matlab toolbox in `PDLF.m`, i.e.

$$\begin{bmatrix} X_j + X_j^T - Y_j & X_j^T H_{F,j} - \begin{bmatrix} Z & 0 \end{bmatrix}^T H_{G,j} \\ H_{F,j} X_j - H_{G,j} \bar{Z} & (1 - \gamma) Y_l \end{bmatrix} > 0,$$

(4.15)

where $H_{F,j}$ and $H_{G,j}$ are the sets of vertices that are contained in $\mathcal{H}_S$ (see Section 4.1.2 and Section 4.1.3), and the state feedback $K = \bar{Z} X_j^{-1}$. If a controller synthesis is desired that is based on a common quadratic LF, we set $Y_j = Y_l = Y$. This is implemented in `CQLF.m`. 

33
4.2 Matlab Implementation With Dropouts

So far, we discussed NCSs that do not incorporate packet dropouts in this chapter. As a consequence, a general model (4.1) is used to describe NCSs and this model can only be analysed on stability using common quadratic Lyapunov functions (discarding the dependence on the overapproximation vertices).

From here, we consider an NCS model that incorporates dropouts as well as time-varying delays and time-varying sampling intervals. In Chapter 3, three different ways to model NCSs that incorporate these network phenomena are proposed. In the Matlab toolbox however, only two modelling approaches are implemented, namely modelling dropouts as a prolongation of the sampling intervals (3.17) (based on [10]) and the explicit dropout modelling using hybrid automata (3.20). For more details about these modelling methods, the reader is referred to Chapter 3. With this extension of the NCS model with dropouts, the only difference with Section 4.1 (no dropouts) is the implementation of the stability analysis in the Matlab toolbox. This is due to the fact that in case of dropouts, the dimension of the set of vertices is depending on the maximum number of subsequent dropouts \( \bar{\delta} \), i.e.

\[
\mathcal{H}_{CL,i,g} = H_{F,i,g} - \begin{bmatrix} K & 0 \end{bmatrix} H_{G,i,g},
\]

for the dropout variable \( i = 0, 1, \ldots, \bar{\delta} \) and the number of vertices \( g = 1, 2, \ldots, 2^{2\nu} \) (in case of modelling dropouts as prolongation of the sampling intervals) or \( i = 0, 1, \text{with} g_0 = 0, 1, \ldots, 2^{2\nu} \) for \( i = 0 \) and \( g_1 = 0, 1, \ldots, 2^{\nu} \) for \( i = 1 \) (in case of modelling dropouts explicitly using hybrid automata). Using this, dropout-dependent LFs can be constructed to analyse the stability of the NCS model. For more details about dropout-dependent LFs, the reader is referred to Section 3.4.

If we use the explicit dropout modelling approach in combination with a PDLF based stability analysis, the following LMIs are used:

\[
P_l > \epsilon I, \\
\mathcal{H}_{CL,0,g_0}^T P_0 \mathcal{H}_{CL,0,g_0} - (1 - \gamma)P_l < 0, \\
\mathcal{H}_{CL,1,g_1}^T P_{j+1} \mathcal{H}_{CL,1,g_1} - (1 - \gamma)P_j < 0,
\]

for all \( j = 0, 1, \ldots, \bar{\delta} - 1 \) and \( l = 0, 1, \ldots, \bar{\delta} \), where the input \( \text{deltaBar} = \bar{\delta} \) and the input \( Dt \) is a cell containing all vertices \( \mathcal{H}_{CL,i,g} \) of the overapproximated NCS model, which are implemented in \([P, LMIs] = \text{LMI(Dt,deltaBar,epsilon)}\).

If we use the dropout as prolongation of the sampling intervals modelling approach in combination with a PDLF based stability analysis, the following LMIs are used:

\[
P_l > \epsilon I, \\
\mathcal{H}_{CL,i,g}^T P_j \mathcal{H}_{CL,i,g} - (1 - \gamma)P_l < 0,
\]

for all \( i, j \in \{0, 1, \ldots, \bar{\delta}\} \) and \( g = 1, 2, \ldots, 2^{2\nu} \), which are implemented in \([P,LMIs] = \text{LMI\_dependent(Dt,deltaBar,epsilon)}\).

If all conditions in (4.17) or (4.18) are satisfied, the analysed NCS is said to be asymptotically stable. If not, no conclusions about the stability of the NCS can be drawn.

4.3 Using the Matlab Toolbox

The user of the Matlab toolbox can choose which of the modelling techniques is used to model an NCS that incorporates time-varying delays, time-varying sampling intervals and packet dropouts. In addition, the user should decide which of the three overapproximation approaches is used to analyse the NCS. This results in six possibilities to start the analysis:

- explicit dropout modelling using hybrid automata with the real Jordan based overapproximation method: Automaton_JNF.m
• explicit dropout modelling using hybrid automata with the Cayley-Hamilton based overapproximation method: **Automaton_CH1.m**

• explicit dropout modelling using hybrid automata with the Cayley-Hamilton based overapproximation method with a reduction of vertices: **Automaton_CH2.m**

• dropout modelling as prolongation of the sampling intervals with the real Jordan based overapproximation method: **Automaton_JNF.m**

• dropout modelling as prolongation of the sampling intervals with the Cayley-Hamilton based overapproximation method: **Automaton_CH1.m**

• dropout modelling as prolongation of the sampling intervals with the Cayley-Hamilton based overapproximation method with a reduction of vertices: **Automaton_CH2.m**

Note that the overapproximation method with a reduction of vertices is implemented using Theorem 2 from [11].

After deciding which techniques are desired in the NCS analysis, one of the six files above is the starting point of the Matlab toolbox and needs the following inputs: $A, B, K, h, \tau, \delta, \epsilon, \gamma$ and the type of Lyapunov function, where

- $A$ and $B$ are the system matrices
- $K$ is the state feedback controller $u_k = -Kx_k$, which can be a full matrix or an empty ($[]$) matrix, if a controller is to be synthesised.
- $h$ is the set of sampling intervals given by $[h_{\text{min}}, h_{\text{max}}]$, where $h_{\text{min}} \leq h_{\text{max}}$.
- $\tau$ is the set of communication delays given by $[\tau_{\text{min}}, \tau_{\text{max}}]$, where $\tau_{\text{min}} \leq \tau_{\text{max}}$.
- $\delta$ is the maximum number of subsequent dropouts that can occur in the NCS.
- $\epsilon$ is a small scalar $0 \leq \epsilon < 1$ that is used to avoid numerical problems while ensuring that the Lyapunov matrix $P$ is positive definite.
- $\gamma$ is a small scalar $0 \leq \gamma < 1$ that ensures a decrease of the LF ($1 - \gamma$ is the convergence rate).
- The type of Lyapunov function input decides what kind of LF is used for the controller synthesis and stability analysis: ‘common’ (CQLF) or ‘pardep’ (PDLF), including the quotation marks.
Chapter 5
Numerical Comparison

To show the differences of the overapproximation techniques that are based on the real Jordan form and Cayley-Hamilton theorem, this chapter discusses a numerical comparison of an NCS analysis in terms of approximation error and conservatism of the stability conditions using three different examples. This also illustrates some of the compatibilities of the designed Matlab toolbox and the theory that is discussed in Chapters 2 and 3. Note that in this chapter we consider NCSs that incorporate constant sampling intervals and time-varying transmission delays without the occurrence of packet dropouts, resulting in a discrete-time model as in (2.4).

Example 5.1
The first example is the double integrator and is taken from [22]. The system is given by (2.1) with
\[
\begin{align*}
A &= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, & B &= \begin{bmatrix} 0 \\ 1 \end{bmatrix}
\end{align*}
\] (5.1)
and the state feedback control law \( u_k = -K x_k \) with \( K = \begin{bmatrix} 6000 & 800 \end{bmatrix} \). In this first example, we take constant sampling intervals \( h = 1 \cdot 10^{-3} \) and time-varying delays with \( \tau_{\text{min}} = 0 \) and \( \tau_{\text{max}} = h = 1 \cdot 10^{-3} \).

Example 5.2
The second example is taken from [33] and is given by (2.1) with
\[
\begin{align*}
A &= \begin{bmatrix} 0 & 1 \\ 0 & -0.1 \end{bmatrix}, & B &= \begin{bmatrix} 0 \\ 0.1 \end{bmatrix}
\end{align*}
\] (5.2)
and the state feedback control law \( u_k = -K x_k \) with \( K = \begin{bmatrix} 3.75 & 11.5 \end{bmatrix} \). In this second example, we take constant sampling intervals \( h = 1 \) and time-varying delays with \( \tau_{\text{min}} = 0 \) and \( \tau_{\text{max}} = 0.1 \).

Example 5.3
The final example is given by (2.1) with
\[
\begin{align*}
A &= \begin{bmatrix} 1 & 15 \\ -15 & 1 \end{bmatrix}, & B &= \begin{bmatrix} 1 \\ 1 \end{bmatrix}
\end{align*}
\] (5.3)
and the state feedback control law \( u_k = -K x_k \) with \( K = \begin{bmatrix} 5.33 & 9.33 \end{bmatrix} \). In this third example, we take constant sampling intervals \( h = 0.01 \) and time-varying delays with \( \tau_{\text{min}} = 0 \) and \( \tau_{\text{max}} = 0.005 \).
5.1 Comparing the Overapproximations

To perform a thorough comparison of the two overapproximation methods based on the real Jordan form and Cayley-Hamilton theorem, the convex overapproximations of $\Gamma := \{ \Gamma(\tau_k) \mid \tau_k \in [\tau_{\min}, \tau_{\max}] \}$ are plotted, where

\[ \Gamma(\tau_k) = \int_0^{h-\tau_k} e^{A_s} ds B, \quad (5.4) \]

with constant sampling intervals $h$ and varying delays $\tau_k$. The reason to plot $\Gamma(\tau_k)$ in (5.4) is that it is the only time-varying function in the NCS model (2.4), when $h_k = h$ is chosen constant. For systems having $x \in \mathbb{R}^2$ and a single input, the function $\Gamma : \mathbb{R}^+ \rightarrow \mathbb{R}^2$ can be plotted. To make the differences between both overapproximation methods visible, we plot $\Gamma(\tau_k)$ together with the overapproximations of $\Gamma(\tau_k)$ as in (4.6) (based on the real Jordan form), i.e.

\[ \Gamma(\tau_k) = S_{3,0} + \sum_{j=1}^{\nu} \alpha_{3,j} S_{3,j} \text{ where } \alpha_{3,j} \in \{ \underline{\alpha}_{3,j}, \bar{\alpha}_{3,j} \}, \quad (5.5) \]

and as in (4.9) (based on the Cayley-Hamilton theorem), i.e.

\[ \Gamma(\tau_k) = \sum_{j=0}^{\nu-1} f_{3,j} A_j B \text{ where } f_{3,j} \in \{ f_{3,j}, \bar{f}_{3,j} \}. \quad (5.6) \]

This results in Figures 5.1(a), 5.2(a) and 5.3(a). These figures show the area of the overapproximations as a measure to indicate the approximation error and therefore conservatism.

Another way to qualify conservatism of the two overapproximation methods is to observe the stabilisation regions of the two methods. Consider Example 5.1 in which the state feedback $u_k = -K x_k$ is used, with $K = \begin{bmatrix} 6000 & K_2 \end{bmatrix}$ (note that $K$ in (2.55) is equal to $\bar{K} = \begin{bmatrix} 6000 & K_2 & 0 \end{bmatrix}$) and the sampling intervals are constant, i.e. $h = 1 \cdot 10^{-3}$. This results in closed-loop systems as in (2.49) parameterised in controller gain $K_2$. In the figures in the next section, the largest $\tau_{\max}$ is plotted as a function of $K_2$ for which the NCS with delays satisfying $\tau_k \in [0, \tau_{\max}]$, for $k \in \mathbb{N}$, is stable. Here, the stability is determined by using common quadratic Lyapunov function based LMIs, as discussed in Section 2.2. In the stabilisation region figures we also displayed the range of the state feedback that result in stable NCSs for constant delays in $[0, \tau_{\max}]$ for references. This forms an upperbound for the stability boundary for state feedbacks for time-varying delays in $[0, \tau_{\max}]$ and is a general indication how conservative the derived stability conditions are with respect to the true stability boundary. The resulting stabilisation regions that result from the real Jordan and Cayley-Hamilton based overapproximation methods are plotted in Figure 5.1(b).

The same is done for Examples 5.2 and 5.3. For Example 5.2 we use the state feedback $u_k = -K x_k$ with $K = \begin{bmatrix} 3.75 & K_2 \end{bmatrix}$, constant sampling $h = 1$ and delays satisfying $\tau_k \in [0, \tau_{\max}]$, for $\tau_{\max} \leq 1$. See Figure 5.2(b) for the results. For Example 5.3 we use a state feedback with $K = \begin{bmatrix} K_1 & 9.33 \end{bmatrix}$, constant sampling $h = 0.01$ and delays satisfying $\tau_k \in [0, \tau_{\max}]$, for $\tau_{\max} \leq 0.01$. The resulting stability regions are given in Figure 5.3(b).

5.2 Results of the Comparison

We can now evaluate the results of the comparison of the two overapproximation methods in terms of the areas of the polytopic embeddings of $\Gamma(\tau_k)$ (5.4) and the stabilisation regions, as discussed in the previous section. In Figure 5.1(a) (for Example 5.1) both overapproximation methods result in the same area (approximately $5 \cdot 10^{-10}$), with the same number of vertices, being $2^2 = 4$. From this, we might expect that the stabilisation regions are similar as well. This expectation is indeed satisfied, as is shown in Figure 5.1(b), where the stabilisation regions for both approaches are the same.
Figure 5.1: Comparison of the real Jordan and Cayley-Hamilton based overapproximation methods using Example 5.1.

For the other two examples, the area of the polytopic embeddings of the two overapproximation techniques are not similar. As is shown in Figure 5.2(a), the Cayley-Hamilton based overapproximation results in a tighter embedding than the real Jordan based method when analysing Example 5.2, i.e. the area of the Cayley-Hamilton approach is approximately $9.1 \cdot 10^{-5}$ and the area of the real Jordan approach is approximately $9.1 \cdot 10^{-4}$. From this observation, we might expect that the Cayley-Hamilton approach results in less conservative stability boundaries than the real Jordan approach. In Figure 5.2(b) is shown that the stabilisation region of the Cayley-Hamilton based method is indeed larger (and thus less conservative) than the real Jordan based method, which meets the expectations.

Figure 5.2: Comparison of the real Jordan and Cayley-Hamilton based overapproximation methods using Example 5.2.
Finally, in Figure 5.3(a) it is shown that for Example 5.3 the real Jordan based overapproximation results in the tightest embedding and from this we might expect that it is therefore less conservative than the Cayley-Hamilton based embedding, i.e. the area of the real Jordan based method is approximately $2.3 \times 10^{-6}$ and the area of the Cayley-Hamilton based method is approximately $5.6 \times 10^{-6}$. However, looking at Figure 5.3(b), we can observe that both overapproximation approaches give similar stability regions. Hence, a smaller real Jordan overapproximation area does not result in less conservative stability boundaries.

Figure 5.3: Comparison of the real Jordan and Cayley-Hamilton based overapproximation methods using Example 5.3.

From the observations that are done in this section can be concluded that in general there is no direct link between the area of the polytopic overapproximation and the stability range. Moreover, we observed that the Cayley-Hamilton based overapproximation technique gives similar or less conservative stability results (i.e. similar or larger stabilisation regions) than the real Jordan based overapproximation, independent of the size of the polytopic overapproximation. This seems to hold true for all three examples in this chapter. Note that, during the analysis of various two-dimensional systems, it was difficult to find a system for which the real Jordan based overapproximation gave a smaller polytopic embedding than the Cayley-Hamilton based approach. From this we can conclude that the Cayley-Hamilton based approach performs best in the used examples, providing a favourable tradeoff between numerical complexity and conservatism of the stability characterisation.
Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this thesis, the stability of networked control systems (NCSs) using a discrete-time model that incorporates time-varying delays, time-varying sampling intervals and packet dropouts is studied. The first goal of this research was to compare different overapproximation techniques numerically in terms of the polytopic overapproximation area and the stabilisation regions. Here, two approaches are considered, namely overapproximations based on the real Jordan form and based on the Cayley-Hamilton theorem. For the used examples, we showed that the Cayley-Hamilton based overapproximation technique results in similar or less conservative stabilisation regions than the real Jordan based method. This observation about stability conservatism seemed to be independent of the size of the polytopic overapproximation. In other words, whether the area of the polytopic embedding resulting from the Cayley-Hamilton based approach is smaller than, larger than or equal to the real Jordan based embedding, the Cayley-Hamilton method gave similar or less conservative stability results for the examples that are used in this thesis. During the analysis of different two-dimensional systems, it was difficult to find a system for which the real Jordan based overapproximation gave a smaller polytopic embedding than the Cayley-Hamilton based approach. It is not clear what causes this difference in polytopic embeddings.

The second goal was to perform a thorough comparison between a selection of existing methods to model dropouts in NCSs. We choose to compare two methods from the literature, namely modelling dropouts as prolongation of the delay and modelling dropouts as prolongation of the sampling interval, with a newly proposed modelling technique that models dropouts explicitly using automata. In the comparison it is shown that the approach of modelling dropouts as prolongation of delays is more conservative than the other two modelling approaches. Moreover, we showed that whether the approach of modelling dropouts as prolongation of the sampling interval in combination with a common quadratic or parameter-dependent Lyapunov function is used or the explicit dropout modelling approach in combination with a parameter-dependent Lyapunov function, both techniques result in comparable stability results. However, using parameter-dependent LFs results in a larger numerical complexity of the stability analysis due to more LMIs. As a consequence, one should make a tradeoff between numerical complexity and stability conservatism.

The third goal of this study was to design a toolbox in Matlab for stability analysis purposes of NCSs. As discussed in Chapter 2, two different dropout modelling approaches, two different overapproximation techniques and a method to synthesise a state feedback are implemented in the Matlab toolbox. The Matlab toolbox enables us to perform the comparisons of the overapproximation and dropout modelling approaches in terms of numerical complexity and conservatism and gives useful illustrative figures. In addition, the Matlab toolbox makes the analysis techniques of NCSs more accessible to other research so that stability of NCSs that incorporate time-varying
or constant sampling intervals, time-varying or constant delays and possible dropouts can now be conveniently analysed.

6.2 Recommendations for Future Research

Still, a lot of research needs to be done to obtain reliable and detailed NCS models that describe all possible network phenomena. First of all, the comparison of the overapproximation techniques needs to be extended with more approaches, such as gridding and Taylor based methods. We studied the overapproximation method based on gridding from [9], which proved to be more conservative and more complex than the other two approaches from this thesis. However, there are more gridding techniques available to perform the overapproximation.

Secondly, this thesis (and most other literature) considers mainly two-dimensional systems. However, higher-order systems need to be studied so that a comparison with the observations on two-dimensional systems can be performed. Note that, the analysis of higher-order systems will result in higher complexities in the overapproximation step as well as in the stability analysis step. As a consequence, the calculation times that are needed to analyse NCSs in the Matlab toolbox will increase. Any knowledge about these calculation times might be useful for a fast and effective way of programming.

For the longer term, other controllers need to be applied to the NCS model, such as dynamic controllers, instead of static state feedback controllers as was done in this thesis. Using other types of controllers will result in different and maybe less conservative stabilisation regions. Also, we considered a discrete-time NCS model that incorporates three network phenomena in this thesis. However, it is possible to model these phenomena in different ways, such as using stochastic modelling. Doing so, will give the possibility to compare the discrete-time modelling approach with other modelling approaches.

Finally, a network not only incorporates varying delays, varying sampling intervals and packet dropouts, but also other phenomena that influence the performance and stability of the NCS, such as quantisation, delays larger than the sampling intervals and communication constraints. By considering all network phenomena simultaneously will result in a complete framework that makes it possible to analyse NCSs more thoroughly.

The ultimate goal is to have a complete Matlab toolbox with a clear user interface, in which one can analyse any NCS that incorporates some or all possible network phenomena by choosing from any desired modelling, overapproximation and stability analysis approach.
Bibliography


