FEDERAL UNIVERSITY OF CEARÁ

CENTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING

POST-GRADUATION PROGRAM IN ELECTRICAL ENGINEERING

MASTER'S PROGRAM IN ELECTRICAL ENGINEERING

JOÃO GABRIEL NAPOLEÃO SILVA

DATA-DRIVEN CONTROL OF NONLINEAR SYSTEMS USING DISSIPATIVITY THEORY

FORTALEZA

2025

JOÃO GABRIEL NAPOLEÃO SILVA

DATA-DRIVEN CONTROL OF NONLINEAR SYSTEMS USING DISSIPATIVITY THEORY

Dissertation submitted to the Master's Program in Electrical Engineering of Post-graduation Program in Electrical Engineering of Center of Technology of Federal University of Ceará, as a partial requirement for obtaining a master's degree in Electrical Engineering. Area of Concentration: Automation on Control

Supervisor: Prof. Dr. Diego de Sousa Madeira

FORTALEZA

2025

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S58d Silva, João Gabriel Napoleão.
Data-driven Control of Nonlinear Systems Using Dissipativity Theory / João Gabriel Napoleão Silva. – 2025.
96 f. : il. color.

Dissertação (mestrado) – Universidade Federal do Ceará, Centro de Tecnologia, Programa de Pós-Graduação em Engenharia Elétrica, Fortaleza, 2025.
Orientação: Prof. Dr. Diego de Sousa Madeira.

1. Controle baseado em dados. 2. Teoria da dissipatividade. 3. Sistemas não-lineares. I. Título.
CDD 621.3

JOÃO GABRIEL NAPOLEÃO SILVA

DATA-DRIVEN CONTROL OF NONLINEAR SYSTEMS USING DISSIPATIVITY THEORY

Dissertation submitted to the Master's Program in Electrical Engineering of Post-graduation Program in Electrical Engineering of Center of Technology of Federal University of Ceará, as a partial requirement for obtaining a master's degree in Electrical Engineering. Area of Concentration: Automation on Control

Approved in: 21/08/2025

EXAMINING BOARD

Prof. Dr. Diego de Sousa Madeira (Supervisor)
Federal University of Ceará (UFC)

Prof. Dr. Wilkley Bezerra Correia
Federal University of Ceará (UFC)

Prof. Dr. Domenico Sgrò
Federal University of Ceará (UFC)

Prof. Dr. Iury Valente de Bessa
Federal University of Amazonas (UFAM)

To my friends, teachers, family and colleagues.

AKNOWLEDGEMENTS

This work was supported by the National Council for Scientific and Technological Development (CNPq), Brazil, under Grant 402731/2023-9 and by the Ceará Foundation to Support Scientific and Technological Development (Funcap)

“People think of education as something they can finish.”

(Isaac Asimov)

RESUMO

No campo de sistemas polinomais, o projeto de controladores comumente consiste em determinar condições de Lyapunov em desigualdades matriciais lineares, e encontrar uma solução através de algoritmos de programação semi-definida. No entanto, as bilinearidades inerentes às condições de Lyapunov exigem empregos de estratégias muitas vezes subótimas, como a iteração D-K, que não possuem critérios de garantia da convergência da solução. Além disso, a obtenção de um modelo que descreva o comportamento de plantas de difícil modelagem é solucionada através de ensaios de coleta de dados para identificação do sistema, os quais, quando sujeitos à perturbações ou ruído, dificultam a obtenção de um modelo confiável. Nesse trabalho, foi utilizada uma abordagem baseada em dados para projetar leis de controle sem um passo intermediário de identificação e com robustez quanto à dados sujeitos a perturbações desconhecidas mas limitadas. Utilizando a teoria de Lyapunov e a teoria da QSR-dissipatividade, foram constuídas condições de soma-de-quadrados para projeto de uma lei de controle para sistemas não-lineares polinomiais que garanta estabilização assintótica localmente em torno da origem. Essas condições foram implementadas utilizando programação em soma-de-quadrados com o auxílio das ferramentas SOSTools e Mosek em algoritmos iterativos sem bilinearidades e que garantissem a convergência da solução. Por fim, a lei de controle e a função de Lyapunov encontrada foram validadas através de ferramentas de verificação formal como o Z3Prover para emitir um certificado de Lyapunov.

Palavras-chave: Controle baseado em dados. Sistemas não-lineares. Teoria da dissipatividade.

ABSTRACT

In the field of polynomial systems, control design is commonly related to finding Lyapunov conditions in the form of linear matrix inequalities and determining a solution to them through semidefinite programming algorithms. However, the bilinearities inherent to Lyapunov conditions require the use of often suboptimal strategies, such as the D-K iteration, which has no guarantees of solution convergence. In addition, the challenge of some plants that are difficult to model is solved through data collection experiments in order to identify the system. Yet, since the data is subject to disturbances or noise, it hinders the development of a reliable model. In this work, a data-driven approach was used to design control laws without an intermediate identification step and with robustness to data subject to unknown but limited disturbances. Using Lyapunov theory and QSR-dissipation theory, sum-of-squares conditions were constructed to design a control law for polynomial nonlinear systems that guarantees asymptotic stabilization locally around the origin. These conditions were implemented using sum-of-squares programming with the help of SOSTools and Mosek in iterative algorithms without bilinearities and constraints for solution convergence. Finally, the control law and the Lyapunov function were validated using formal verification tools such as Z3Prover to issue a Lyapunov certificate.

Keywords: Data-driven control. Nonlinear systems. Dissipativity theory.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

CEGIS	Counter-Example Guided Inductive Synthesis
RDBI	Recurrent Dissipativity-Based Inequalities
SMT	Satisfiability Modulo Theories
SOF	Static Output Feedback
SOS	Sum-Of-Squares
SOSP	Sum-of-Squares Program

LIST OF SYMBOLS

$A^\top$	Transpose of matrix A.
I	Identity matrix.
$J_{m\times n}$	$m \times n$ matrix of ones.
$diag\{a_1, \cdots, a_n\}$	Diagonal n×n matrix with $a_1, \cdots, a_n$ in its main diagonal.
$ker(A)$	Kernel of a matrix operator.
$\forall x$	For all values of x.
$A \succ 0$	Positive definite matrix.
$A \succeq 0$	Positive semidefinite matrix.
$A \prec 0$	Negative definite matrix.
$A \preceq 0$	Negative semidefinite matrix.
$\nabla\phi(x)$	Gradient of $\phi(x)$, given by $\left[\frac{\partial\phi}{\partial x_1}\ \frac{\partial\phi}{\partial x_2}\cdots\frac{\partial\phi}{\partial x_n}\right]^T$
$\Sigma\left[x\right]$	Sum of squares of x.
$\|\|x\|\|$	Euclidean norm of vector x, given by $\sqrt{x^\top x}$.
$\mathbb{N}$	Set of natural numbers.
$\mathbb{R}$	Set of real numbers.
$\mathbb{R}^{m\times n}$	Set of real $m \times n$ matrices.
$\mathbb{S}^m$	Set of symmetrical $m \times m$ matrices.
$\mathbb{S}^m_{>0}$	Set of positive definite $m \times m$ matrices.
$\mathscr{P}^{m\times n}$	Set of polynomial $m \times n$ matrices.
$\langle u, v\rangle$	Inner product between vectors u and v, equal to $u^\top v$.
$\square$	QED - *Quod erat demonstrandum*/That which was to be demonstrated.

SUMMARY

1 INTRODUCTION

The control of dynamical systems is a widely present task in all engineering fields. Beacause of that, the study of such systems is of utmost importance, since the modeling through mathematical equation makes it possible to analyze its behaviour and evolution over time (HADDAD; CHELLABOINA, 2008). One possible way to describe the systems in the time domain is the state-space representation, which utilizes first order differential equations of the derivatives of the variables of interest (denominated *states*) in a system of equations, allowing, given an initial value, the determination of the states in any point of time. in order to implement a control system, it is possible to describe the system with the desired degree of accuracy around an operation point. Such control system may be responsible for guaranteeing the desired behavior, determined by each application, and its performance criteria, which could be, e.g, stabilization around an equilibrium point, disturb rejection, or reference tracking.

Dynamical systems in general are nonlinear, due to their own nature or due to the nature of the control system implemented. The presence of nonlinearities can turn the analysis of the systems more difficult and, for that reason, the system is linearized around a point of interest to circumvent such issue. Linearizing a system can be sufficient in many cases, and it has advantages such as making stability analysis easiest, being sufficient to check the signal of state-space matrix eigenvalues. However, it limits the capacity of the model to describe the real system to a neighborhood of the selected operation point, which can be a very small domain of model-system equivalence, making necessary an alternative model that takes into account the nonlinearities inherent to the studied system.

Lyapunov stability theory is an important and well-known tool for the study of dynamical systems and can be applied to nonlinear systems, since it provides conditions that, if satisfied, guarantee different degrees of stability of an equilibrium point (KHALIL, 2002). With such conditions, the procedure of stabilization of a plant around the origin can become the design of a Static Output Feedback (SOF) control law that makes the closed-loop system stable, asymptotically stable, or exponentially stable by satisfying the respective Lyapunov conditions. In the recent years, Lyapunov theory is being applied in fields such as stabilization of polynomial systems using Sum-Of-Squares (SOS) decomposition and Dissipativity theory. Additionally, the development of softwares such as YALMIP (LÖFBERG, 2004), SOSTools (PAPACHRISTODOULOU *et al.*, 2021), SeDuMi (STURM, 1999) and MOSEK (APS, 2025) has made possible to development of algorithms that solve these problems numerically. In this

scenario, (AUGUST; PAPACHRISTODOULOU, 2022) develops an algorithm based in Sum-of-Squares Program (SOSP) to design SOF control laws for polynomial systems by imposing constraints in the Lyapunov function and state-space model. Such constraints result in a linear-like approach to the state-space polynomial model. In (ICHIHARA, 2013), a more general assumption in the Lyapunov function is made, being that a quadratic polynomial structure, and a solution for symmetric input saturation is proposed using polytopic conditions, although still dealing with a linear-like state-space model. As for methods of verifying the Lyapunov conditions in dynamical systems, (AHMED *et al.*, 2020) uses a strategy of Counter-Example Guided Inductive Synthesis (CEGIS) to synthesize a Lyapunov function for a given system with polynomial constraints with a Satisfiability Modulo Theories (SMT) solver. In addition, (ABATE *et al.*, 2021) applies the same strategy but considers a neural Lyapunov function.

Although modelling a system with only phenomenological theory can be sufficient in many applications, it results in deviations between the model and the real system due to, e.g., precision or tolerances. In this scenario, some strategies rely on some tests and experiments in order to collect data from the plant. Then, the data can be used to construct a model for the plant by applying identification techniques, which is used later on in the control design methods. Alternatively, an innovative set of strategies search to design control laws directly from the data, without the intermediary step of identifying the system. In (WILLEMS *et al.*, 2005), it is shown that if the input-output data is persistently exciting, it can represent a linear system's whole behaviour. In (MARTIN; ALLGöWER, 2024), this notion is utilized to verify dissipativity conditions from data in nonlinear systems through Taylor's polynomial approximation. (PERSIS; TESI, 2019) also uses Willems' lemma as a primary assumption to design data-driven control laws with linear matrix inequalities. The same issue is addressed in (BISOFFI *et al.*, 2022) with the addition of a solution for polynomial systems with no constraints in Lyapunov function aside it being polynomial. This solution, in order to circumvent the bilinearities in the SOS constraints, uses SOSP to solve an algorithm with D-K iteration (an iterative method that alternately fixes one of the terms of a bilinearity to solve the problem for the other decision variable), with stop criterion of maximum iterations.

In this scenario, the objective of this work will be the construction of algorithms that have no constraint in the structure of the Lyapunov function, besides it being polynomial, and have a stop criterion that guarantees a valid solution was found. By using SOS decomposition to design control laws based in the QSR-Dissipativity theory and Lyapunov theory, with a

data-driven approach, the designed control law must satisfy Lyapunov conditions for local asymptotical stabilization around the origin and it will be explored the impact of different definitions for the supply rate present in the QSR-Dissipativity theory. Furthermore, it will also be explored the certification of Lyapunov functions by the SMT solver: Z3 Theorem Prover, a formal verification tool (MOURA; BJøRNER, 2008).

2 THEORETICAL BASIS

In the field of control of dynamical systems, the modelling process is of utmost importance for the analysis of the systems' behaviour and the design of control laws that assure the desired performance criteria. With that purpose, dynamical systems will be studied in this work from their state space representation, since it allows to determine the trajectories of the states of the system over time given a set of first order differential equations and a set of initial values (HADDAD; CHELLABOINA, 2008).

2.1 Dynamical Systems

To write this definition formally, consider a state vector $x \in \mathscr{X} \subseteq \mathbb{R}^n$ and an input vector $u \in \mathscr{U} \subseteq \mathbb{R}^m$. A *time-varying* nonlinear system can be written as:

$$\dot{x} = F(t, x(t), u(t)), \quad x(t_0) = Z_0, \quad x \in [t_0, t_1]. \tag{2.1}$$

The system is said to be *time-invariant* if

$$F(t_0, x(t), u(t)) = F(t, x(t), u(t)), \quad \forall (t, x, u) \in [t_0, t_1] \times \mathscr{X} \times \mathscr{U}. \tag{2.2}$$

If the system has the form:

$$\dot{x} = f(x) + g(x)u, \tag{2.3}$$

it is classified as *input-affine*. Here, a subtype of such systems will be studied, the *polynomial* systems, in which $f(x)$ e $g(x)$ can be written as polynomials of x (BISOFFI *et al.*, 2022). Therefore, they can be modelled using matrices of monomials $Z(x) \in \mathscr{P}^{N\times 1}$ e $W(x) \in \mathscr{P}^{M\times m}$ and matrices of coefficients $A \in \mathbb{R}^{n\times N}$ e $B \in \mathbb{R}^{n\times M}$, such that,

$$\dot{x} = AZ(x) + BW(x)u. \tag{2.4}$$

Additionally, an output $y \in \mathscr{Y} \subseteq \mathbb{R}^p$ can be defined such that

$$y = h(x). \tag{2.5}$$

The equation pair (2.2) and (2.5) (or (2.4) and (2.5)) compose the state-space representation of a nonlinear (polynomial) system. Since innumerable applications aim to achieve stabilization around an equilibrium point, its definition becomes necessary.

2.2 Equilibrium Points

An stable equilibrium point x_{eq}, considering $x \in \mathscr{X}$, is the one that $x \to x_{eq}$ when $t \to \infty$. This point is considered *isolated* if there is an open ball around it, defined by $B_\varepsilon(x_{eq}) := \{x \in \mathbb{R}^n : ||x - x_{eq}|| < \varepsilon\}$, which contains only that equilibrium point.

To study the system's behaviour around the equilibrium point, the Lyapunov Theory will be used as in (HADDAD; CHELLABOINA, 2008).

2.3 Lyapunov Stability Theory

Lyapunov theory analyzes the stability around an equilibrium point (specifically here, around the origin) by finding a Lyapunov function that attests the stability of a system to a certain degree. These degrees of stability are dependants on the conditions met by the Lyapunov function, and to define them consider the nonlinear dynamical system around the origin:

$$\dot{x} = f(x(t)), \quad x(0) = Z_0, \quad x(t) \in \mathscr{X}, \quad 0 \in \mathscr{X}. \tag{2.6}$$

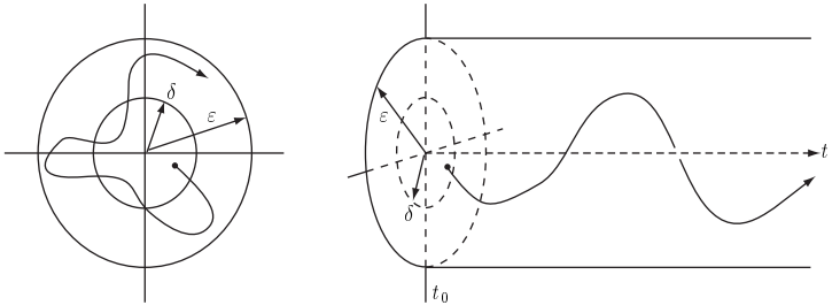
Assume $f(0) = 0$ and $f(x)$ being Lipschitz continuous, i.e., $||f(x) - f(y)|| \leq L||x - y||$, $\forall x, y$, for some $L > 0$. Such assumption implies that there is a different $x(t)$ for each initial condition of the system and that it is solution of (2.6). Then, the stability of Lyapunov therory is defined as:

Definition 1. *(i) The solution $x(t) \equiv 0$ of (2.6) is Lyapunov stable if, for every $\varepsilon > 0$, there is a $R_D(\varepsilon) > 0$ such that, if $||x(0)|| < R_D$, then $||x(t)|| < \varepsilon$, $t \geq 0$.*

(ii) The solution $x(t) \equiv 0$ of (2.6) is asymptotically stable if it is Lyapunov stable and there is a $R_D > 0$ such that, if $||x(0)|| < R_D$, then $\lim_{x\to\infty} x(t) = 0$.

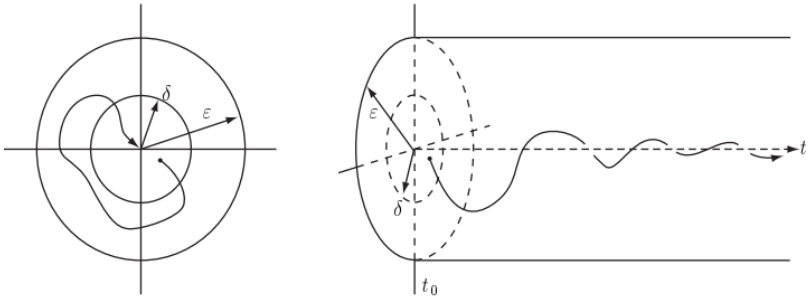
(iii) The solution $x(t) \equiv 0$ of (2.6) is exponentially stable if there are α, β e R_D constants such that, if $||x(0)|| < R_D$, then $||x(t)|| \leq \alpha||x(0)||e^{-\beta t}$, $t \geq 0$.

Figure 1 – Lyapunov stability



Source: (HADDAD; CHELLABOINA, 2008)

Figure 2 – Asymptotical stability



Source: (HADDAD; CHELLABOINA, 2008)

In figure 1, note that Lyapunov stability does not necessarily implies $x \to x_{eq}$, although it implies in x having a maximum distance radius from the origin. In figure 2 however, the asymptotical stability implies $x \to x_{eq}$ when $t \to \infty$.

In order to verify the degree of stability of the analyzed system according to Definition 1, Lyapunov Theorem is stablished as it follows:

Theorem 1 (Lyapunov Theorem)**.** *Consider the nonlinear dynamical system (2.6) and assume that there is a continuously differentiable function* $V : \mathscr{X} \to \mathbb{R}$ *such that*

$$\begin{aligned} &V(0) = 0, \\ &V(x) > 0, \qquad x \in \mathscr{X},\ x \neq 0, \\ &\nabla V(x)^\top f(x) \leq 0, \qquad x \in \mathscr{X}, \end{aligned} \tag{2.7}$$

then the solution $x(t) \equiv 0$ *of (2.6) is Lyapunov stable. If, in addition to that,*

$$\nabla V(x)^\top f(x) < 0, \quad x \in \mathscr{X}, x \neq 0, \tag{2.8}$$

then the solution $x(t) \equiv 0$ *of (2.6) is asymptotically stable. At last, if there are* $\alpha, \beta, \varepsilon > 0$ *and*

$p \geq 1$, such that

$$\begin{aligned} \alpha||x||^p \leq V(x) \leq \beta||x||^p, \quad x \in \mathscr{X}, \\ \nabla V(x)^\top f(x) \leq -\varepsilon V(x), \quad x \in \mathscr{X}, \end{aligned} \tag{2.9}$$

then the solution $x(t) \equiv 0$ of (2.6) is exponentially stable.

In control theory, even if the open-loop system is not Lyapunov, asymptotically or exponentially stable, one can guarantee the stability of the closed-loop system by taking into account the conditions of Theorem 1 when designing the control system.

2.4 Dissipativity Theory

Another important theory that will be applied further in this work is the Dissipativity Theory (WILLEMS, 2007), which will be implemented alongside Lyapunov theorem. In order to do so, considering the input-affine system (2.3)-(2.5), a *supply rate* function will be defined as:

$$r(u, y(x)) = y(x)^\top Q y(x) + 2y(x)^\top S u + u^\top R u, \tag{2.10}$$

where $Q \in \mathbb{S}^p$, $S \in \mathbb{R}^{p \times m}$, $R \in \mathbb{S}^m_{>0}$. A system is called to be *dissipative* (or *QSR-dissipative* for the supply rate chosen here) if there exists a storage function $V(x)$ such that

$$\dot{V}(x) \leq r(u, y(x)). \tag{2.11}$$

Additionally, the system can be called *strictly QSR-dissipative* if it satisfies

$$\dot{V}(x) + T(x) \leq r(u, y(x)), \tag{2.12}$$

for some $T(x) > 0$. This concept of dissipative systems can be taken into account when designing a control law for a SOF by selecting $u = -R^{-1}S^\top y(x)$, meaning that (2.10) can be developed as it follows:

$$r(u, y(x)) = y(x)^\top Q y(x) + 2y(x)^\top S u + u^\top R u \tag{2.13}$$

$$r(u, y(x)) = y(x)^\top Q y(x) - 2y(x)^\top S R^{-1} S^\top y(x) + y(x)^\top S R^{-1} S^\top y(x) \tag{2.14}$$

$$r(u, y(x)) = -y(x)^\top (S R^{-1} S^\top - Q) y(x) \tag{2.15}$$

Defining a Δ_c as

$$\Delta_c := S R^{-1} S^\top - Q, \tag{2.16}$$

and if $\Delta_c > 0$, then $r(u, y(x)) \leq 0$, which in turn implies $\dot{V} < 0$, satisfying the asymptotical stability conditions in Theorem 1.

2.5 Data-driven control

Designing a control law through modeling is not always possible or viable. This can occur, e.g., when there are parameters difficult to calculate or not available for measuring. In such cases, the plant can be considered a black box, when there is no prior knowledge of its dynamics, or a gray box, when some characteristics, such as degree of polynomial complexity, is known beforehand. Then, an experiment can be performed to gather data with enough information to describe the plant behavior. Here, it is considered a unknown process disturbance d which affects the evolution of the system resulting in noisy data. For that reason, it is necessary to take into account a set of systems compatible with the data, in order to obtain a robust control law.

2.5.1 Noisy data and uncertainty set $\mathscr{C}$

In (BISOFFI *et al.*, 2022), when describing the dynamics of the systems using data, its representation can be derived from (2.4) as

$$\dot{x} = A_* Z(x) + B_* W(x) + d, \quad x \in \mathbb{R}^n,\ u \in \mathbb{R}^m,\ d \in \mathbb{R}^n. \tag{2.17}$$

$Z(x)$ and $W(x)$ are considered the **known** vector and matrix of regressors, respectively. A_* and B_* are **unknown** matrices containing the coefficients of the regressors from the exact plant model. In this arrangement, Z and W can be chosen containing more monomials than the ones present in the plant, since the coefficients from the extra monomials would ideally be null. A noisy-data is considered in this model, and the disturbance d represents the measurement noises that are unavoidable in the data-acquiring proccesses. The experiment, necessary to obtain the system's data since A_* and B_* are unknown, is performed over $t \in [t_0, t_{T_s-1}]$ (being T_s the number of samples by applying an input sequence $u_{t_0},\ \ldots,\ u_{T_s-1}$ and collecting the input-state sequences in the data-matrices $X_1 \in \mathbb{R}^{n\times T_s}$, $Z_0 \in \mathbb{R}^{N\times T_s}$, $W_0 \in \mathbb{R}^{M\times T_s}$, defined as:

$$X_1 := \begin{bmatrix} \dot{x}(t_0) & \cdots & \dot{x}(t_{T_s-1}) \end{bmatrix}, \tag{2.18a}$$

$$Z_0 := \begin{bmatrix} Z(x(t_0)) & \cdots & Z(x(t_{T_s-1})) \end{bmatrix}, \tag{2.18b}$$

$$W_0 := \begin{bmatrix} W(x(t_0))u(t_0) & \cdots & W(x(t_{T_s-1}))u(t_{T_s-1}) \end{bmatrix}. \tag{2.18c}$$

The disturbance sequence cannot be measured, but it is already contained in the data in (2.18). For that reason and in order to robustly model the problem, it is considered a disturbance sequence

with bounded energy, i.e., for some matrix R_D:

$$D_0 := \begin{bmatrix} d(t_0) & \cdots & d(t_{T_s-1}) \end{bmatrix} \tag{2.19a}$$

$$D_0 \in \mathscr{D} := \{D \in \mathbb{R}^{n \times T_s} : DD^\top \preceq R_D R_D^\top\}. \tag{2.19b}$$

Together, (2.18) and (2.19) make up the *uncertainty set* $\mathscr{C}$:

$$\mathscr{C} := \{[A \;\; B] : X_1 = AZ_0 + BW_0 + D, \; D \in \mathscr{D}\} \tag{2.20}$$

which describes all $[A \;\; B]$ matrix pairs *consistent with data*, i.e., that could generate the sequences in (2.18)

2.5.2 Reformulations of $\mathscr{C}$ and properties

In this subsection, the aim is to reformulate the set $\mathscr{C}$ in (2.20) as a matrix ellipsoid in order to derive SOS constraints for control design.

By isolating D in set $\mathscr{C}$ definition (2.20), one obtains $D = X_1 - AZ_0 - BW_0$, and substituting it in (2.19b) results in

$$(X_1 - AZ_0 - BW_0)(X_1 - AZ_0 - BW_0)^\top \preceq R_D R_D^\top.$$

The previous expression can be developed and reorganized in a quadratic form following the next steps:

$$\begin{gathered}
X_1X_1^\top - X_1Z_0^\top A^\top - X_1W_0^\top B^\top - AZ_0X_1^\top + AZ_0Z_0^\top A^\top \\
+AZ_0W_0^\top B^\top - BW_0X_1^\top + BW_0Z_0^\top A^\top + BW_0W_0^\top B^\top - R_DR_D^\top \preceq 0 \\
\begin{bmatrix} 1 & A & B \end{bmatrix} \begin{bmatrix} X_1X_1^\top - R_DR_D^\top & \begin{bmatrix} -X_1W_0^\top & -X_1W_0^\top \end{bmatrix} \\ \begin{bmatrix} -Z_0X_1^\top \\ -W_0X_1^\top \end{bmatrix} & \begin{bmatrix} Z_0Z_0^\top & Z_0W_0^\top \\ -W_0Z_0^\top & W_0W_0^\top \end{bmatrix} \end{bmatrix} \begin{bmatrix} 1 \\ A^\top \\ B^\top \end{bmatrix} \preceq 0 \\
\begin{bmatrix} 1 & A & B \end{bmatrix} \begin{bmatrix} X_1X_1^\top - R_DR_D^\top & -X_1 \begin{bmatrix} Z_0^\top & W_0^\top \end{bmatrix} \\ -\begin{bmatrix} Z_0 \\ W_0 \end{bmatrix} X_1^\top & \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix} \begin{bmatrix} Z_0^\top & W_0^\top \end{bmatrix} \end{bmatrix} \begin{bmatrix} 1 \\ A^\top \\ B^\top \end{bmatrix} \preceq 0 \\
\begin{bmatrix} 1 & A & B \end{bmatrix} \begin{bmatrix} X_1X_1^\top - R_DR_D^\top & -X_1 \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix}^\top \\ -\begin{bmatrix} Z_0 \\ W_0 \end{bmatrix} X_1^\top & \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix} \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix}^\top \end{bmatrix} \begin{bmatrix} 1 \\ A^\top \\ B^\top \end{bmatrix} \preceq 0
\end{gathered}$$

This matricial product can be written in terms of new data matrices A_d, B_d, and C_d. Defining them as

$$\begin{bmatrix} C_d & B_d^\top \\ B_d & A_d \end{bmatrix} := \begin{bmatrix} X_1X_1^\top - R_DR_D^\top & -X_1 \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix}^\top \\ -\begin{bmatrix} Z_0 \\ W_0 \end{bmatrix} X_1^\top & \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix} \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix}^\top \end{bmatrix} \tag{2.21}$$

allows the set $\mathscr{C}$ to be expressed in terms of A_d, B_d, and C_d, expressed below.

$$\begin{aligned} \mathscr{C} &= \{[A \;\; B] : [1 \;\; A\; B] \begin{bmatrix} C_d & B_d^\top \\ B_d & A_d \end{bmatrix} [1 \;\; A\; B]^\top \preceq 0\} \\ &= \{[A \;\; B] = Z_{AB}^\top : C_d + B_d^\top Z_{AB} + Z_{AB}^\top B_d + Z_{AB}^\top A_d Z_{AB} \preceq 0\} \end{aligned} \tag{2.22}$$

The data matrices A_d, B_d, C_d are not related to an identification step for the real system $[A_* \; B_*]$, but are parameters for estimating a set containing all pairs $[A \;\; B]$ consistent with data. Taking in consideration a standard representation of a vector ellipsoid, i.e., $\{z \in \mathbb{R}^q : c + b^\top z + z^\top b + z^\top a z \leq 0\}$ with parameters $a \in \mathbb{R}^{q\times q}$, $b \in \mathbb{R}^q$, and $c \in \mathbb{R}$, it is notable that (2.21) can be interpreted as an extension for matrix ellipsoids. In this interpretation, and since $A_d = A_d^\top$ in (2.21), (2.22) is further developed in the next steps by multiplying terms equivalent with the identity matrix and adding a term equivalent to zero.

$$\begin{gathered} C_d + B_d^\top Z_{AB} + Z_{AB}^\top B_d + Z_{AB}^\top A_d Z_{AB} \preceq 0 \\ C_d + (A_dA_d^{-1}B_d)^\top Z_{AB} + Z_{AB}^\top A_dA_d^{-1}B_d + Z_{AB}^\top A_dZ + B_d^\top A_d^{-1}A_dA_d^{-1}B_d - B_d^\top A_d^{-1}A_dA_d^{-1}B_d \preceq 0 \\ C_d + (A_d^{-1}B_d)^\top A_dZ_{AB} + Z_{AB}^\top A_dA_d^{-1}B_d + Z_{AB}^\top A_dZ_{AB} + (A_d^{-1}B_d)^\top A_dA_d^{-1}B_d - B_d^\top A_d^{-1}B_d \preceq 0 \\ (A_d^{-1}B_d + Z_{AB})^\top A_d(A_d^{-1}B_d + Z_{AB}) \preceq B_d^\top A_d^{-1}B_d - C_d \end{gathered}$$

Defining the matrices $\zeta := -A_d^{-1}B_d$ and $Q_d := B_d^\top A_d^{-1}B_d - C_d$, one obtains a new representation of set $\mathscr{C}$.

$$\begin{gathered} \mathscr{C} = \{[A \;\; B] = Z_{AB}^\top : (Z_{AB} - \zeta)^\top A_d (Z_{AB} - \zeta) \preceq Q_d\} \\ \zeta = -A_d^{-1}B_d, \quad Q_d = B_d^\top A_d^{-1}B_d - C_d \end{gathered} \tag{2.23}$$

Note that this new representation of the matrix ellipsoid is defined around its center ζ with its radius being Q_d, which will be particularly useful when designing a control law that stabilizes the whole set of matrices consistent with the data obtained. This will guarantee some level of

robustness related to the noise and its assumed upper bound R_D, taken into account explicitly in the data matrix C_d (2.21).

For the next lemmas, it will be assumed that the data $\left[\begin{smallmatrix} Z_0 \\ W_0 \end{smallmatrix}\right]$ is *persistently exciting* (WILLEMS *et al.*, 2005), which is related to its *full row rank*. The implication is that the data will have enough information to represent the behavior of the system in a way possible to check whether or not one needs more samples. The persistence of excitation directly implies that $A_d \succ 0$, which guarantees it is an invertible matrix, necessary for the definition (2.23). Also, since Q_d represents an ellipsoid radius, it needs to be positive semidefinite, a result that will be proven in Lemma 1.

Lemma 1. *Assuming* $\left[\begin{smallmatrix} Z_0 \\ W_0 \end{smallmatrix}\right]$ *has full row rank,* $A_d \succ 0$ *and* $Q_d \succeq 0$.

Proof. $A_d \succ 0$ is a direct consequence of $\left[\begin{smallmatrix} Z_0 \\ W_0 \end{smallmatrix}\right]$ having full row rank (BISOFFI *et al.*, 2022), with A_d as in (2.21). Q_d can be developed by substituting (2.21) in (2.23):

$$\begin{aligned} Q_d &= B_d^\top A_d^{-1} B_d - C_d \\ &= X_1 \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix}^\top \left(\begin{bmatrix} Z_0 \\ W_0 \end{bmatrix} \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix}^\top \right)^{-1} \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix} X_1^\top - X_1 X_1^\top + R_D R_D^\top \end{aligned}$$

Defining $Q_p = \left[\begin{smallmatrix} Z_0 \\ W_0 \end{smallmatrix}\right]^\top \left(\left[\begin{smallmatrix} Z_0 \\ W_0 \end{smallmatrix}\right] \left[\begin{smallmatrix} Z_0 \\ W_0 \end{smallmatrix}\right]^\top \right)^{-1} \left[\begin{smallmatrix} Z_0 \\ W_0 \end{smallmatrix}\right]$, it can be verified that $Q_p^2 = Q_p$, which qualifies Q_p as a projection matrix, and the previous equation can be written as

$$Q_d = X_1 Q_p X_1^\top - X_1 X_1^\top + R_D R_D^\top. \tag{2.24}$$

Additionally, being $X_1 = A_* Z_0 + B_* W_0 + D_0 = \left[\begin{smallmatrix} A_* & B_* \end{smallmatrix}\right] \left[\begin{smallmatrix} Z_0 \\ W_0 \end{smallmatrix}\right] + D_0$, it follows that

$$
\begin{aligned}
X_1 Q_p - X_1 = &\left(\begin{bmatrix} A_* & B_* \end{bmatrix} \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix} + D_0 \right) \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix}^\top \left(\begin{bmatrix} Z_0 \\ W_0 \end{bmatrix} \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix}^\top \right)^{-1} \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix} \\
&- \begin{bmatrix} A_* & B_* \end{bmatrix} \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix} - D_0 \\
\\
&= \begin{bmatrix} A_* & B_* \end{bmatrix} \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix} + D_0 \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix}^\top \left(\begin{bmatrix} Z_0 \\ W_0 \end{bmatrix} \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix}^\top \right)^{-1} \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix} \\
&- \begin{bmatrix} A_* & B_* \end{bmatrix} \begin{bmatrix} Z_0 \\ W_0 \end{bmatrix} - D_0 \\
\\
&= D_0 Q_p - D_0.
\end{aligned}
$$

A similar result for its transpose can be obtained as well: $Q_p X_1^\top - X_1^\top = (X_1 Q_p - X_1)^\top = (D_0 Q_p - D_0)^\top = Q_p D_0^\top - D_0^\top$. At last, (2.24) can be rearranged in order to use these results, allowing to write it in terms of D_0 instead of X_1, by the next steps.

$$
\begin{aligned}
Q_d &= X_1 Q_p X_1^\top - X_1 X_1^\top + R_D R_D^\top \\
&= (X_1 Q_p - X_1) X_1^\top + R_D R_D^\top \\
&= (D_0 Q_p - D_0) X_1^\top + R_D R_D^\top \\
&= D_0 (Q_p X_1^\top - X_1^\top) + R_D R_D^\top \\
&= D_0 (Q_p D_0^\top - D_0^\top) + R_D R_D^\top \\
&= D_0 Q_p D_0^\top - D_0 D_0^\top + R_D R_D^\top
\end{aligned}
$$

Since $Q_p \succeq 0$ and $D_0 \in \mathscr{D}$ (2.19b),

$$
\begin{aligned}
Q_d &= D_0 Q_p D_0^\top + R_D R_D^\top - D_0 D_0^\top \\
&\succeq R_D R_D^\top - D_0 D_0^\top \\
&\succeq 0.
\end{aligned}
$$

□

Lemma 2. *Assuming* $\left[\begin{smallmatrix} Z_0 \\ W_0 \end{smallmatrix}\right]$ *has full row rank,* $\mathscr{C}$ *is bounded with respect to any matrix norm.*

Proof. Consider $\mathscr{C}$ in (2.23), a nonempty set since $\zeta^\top \in \mathscr{C}$. $Z_{AB}^\top \in \mathscr{C}$ if and only if $v^\top (Z_{AB} - \zeta)^\top A_d (Z_{AB} - \zeta) v \leq v^\top Q_d v$, for all $v \in \mathbb{R}^n$. $\lambda_{min}(A_d)$ represents the minimum eigenvalue of A_d (symmetric). By applying the square root in both sides of the previous inequation, one obtains

$$|A_d^{1/2}||(Z_{AB} - \zeta)v| \leq |Q_d^{1/2} v|, \quad \forall v : |v| = 1.$$

Using the induced 2-norm, as in $|A_d^{1/2}|^2 = \lambda_{min}((A_d^{1/2})^\top A_d^{1/2}) = \lambda_{min}(A_d)$ e $|Mv| = \sup_{|v|=1} |Mv|$ (HORN; JOHNSON, 2013), implies

$$\sqrt{\lambda_{min}(A_d)} \sup_{|v|=1} |(Z_{AB} - \zeta)v| \leq \sup_{|v|=1} |Q_d^{1/2} v|.$$

Through the reverse triangle inequality, it is possible to obtain the following development.

$$\begin{aligned}
&\sup_{|v|=1} |(Z_{AB} - \zeta)v| \leq \lambda_{min}(A_d)^{-1/2} \sup_{|v|=1} |Q_d^{1/2} v| \\
&\implies ||Z_{AB} - \zeta|| \leq \lambda_{min}(A_d)^{-1/2} ||Q_d^{1/2}|| \\
&\implies ||Z_{AB}|| - ||\zeta|| \leq ||Z_{AB} - \zeta|| \leq \lambda_{min}(A_d)^{-1/2} ||Q_d^{1/2}|| \\
&\implies ||Z_{AB}|| \leq ||\zeta|| + \lambda_{min}(A_d)^{-1/2} ||Q_d^{1/2}||
\end{aligned}$$

Since all the values from the right side of the inequation are finite, Z is bounded in relation to the 2-norm. Furthermore, any matrix norm is equivalent to another matrix norm, meaning there is a finite constant C_{AB}, such that $||M||_a \leq C_{ab} ||M||_b$ to any matrix M (HORN; JOHNSON, 2013). Thus, if Z_{AB} is bounded in relation to 2-norm, implies it being bounded in relation to any n-norm. □

In order to apply Petersen's Lemma, which will be presented further in this work, it is convenient to define set $\mathscr{C}$ as set $\mathscr{E}$ will, defined by:

$$\mathscr{E} := \{\zeta + A_d^{-1/2} \Upsilon Q_d^{1/2} : ||\Upsilon|| \leq 1\}. \tag{2.25}$$

Given a matrix inequality with a bounded matrix F (i.e, $F^\top F \preceq \bar{F}$), Petersen's lemma allows to obtain an equivalent condition in terms of its upper bound $\bar{F}$. In the case of set $\mathscr{E}$, by taking $F = \Upsilon$ and $\bar{F} = I$, Petersen's lemma will be useful when obtaining conditions for control design in terms of the data. Before that, the next proposition will show that set $\mathscr{C}$ is equivalent to set $\mathscr{E}$.

Proposition 1. *Being $A_d \succ 0$ and $Q_d \succeq 0$, $\mathscr{C} = \mathscr{E}$.*

Proof. To prove that $\mathscr{C} = \mathscr{E}$, it will be proved that $\mathscr{C} \subseteq \mathscr{E}$ and $\mathscr{E} \subseteq \mathscr{C}$.

For $\mathscr{C} \subseteq \mathscr{E}$, suppose that $Z_{AB}^{\top} \in \mathscr{E}$, i.e., $Z_{AB}^{\top} = \zeta + A_d^{-1/2}\Upsilon Q^{1/2}$, for some matrix Υ satisfying $||\Upsilon|| \leq 1$. Then,

$$\begin{aligned}(Z_{AB}-\zeta)^{\top}A_d(Z_{AB}-\zeta) &= (A_d^{-1/2}\Upsilon Q^{1/2})^{\top}A_d(A_d^{-1/2}\Upsilon Q^{1/2}) \\ &= Q^{1/2}\Upsilon^{\top}A_d^{-1/2}A_dA_d^{-1/2}\Upsilon Q^{1/2} \\ &= Q^{1/2}\Upsilon^{\top}\Upsilon Q^{1/2} \\ &\preceq Q^{1/2}Q^{1/2} = Q.\end{aligned}$$

For $\mathscr{C} \subseteq \mathscr{E}$, suppose that $Z_{AB}^{\top} \in \mathscr{C}$, i.e., $(Z_{AB}-\zeta)^{\top}A_d(Z_{AB}-\zeta) \preceq Q$. A matrix Υ needs to be found, satisfying $Z_{AB} = \zeta + A_d^{-1/2}\Upsilon Q^{1/2}$, which can be written as

$$\Upsilon Q^{1/2} = A_d^{1/2}(Z_{AB}-\zeta). \tag{2.26}$$

If $Q^{1/2} = 0$, this condition can be satisfied by taking $\Upsilon = 0$. If $Q^{1/2} \succ 0$, it has $i \in \{1,\cdots,n\}$ that define a diagonal matrix $\Lambda_i := diag\{\lambda,\cdots,\lambda_i\} \succ 0$. Since $Q^{1/2}$ is symmetrical, there is an orthogonal real matrix T ($T^{\top}T = TT^{\top} = I$), such that the eigenvalues decomposition of $Q^{1/2}$ is

$$Q^{1/2} = T\Lambda T^{\top} := T\left[\begin{smallmatrix}\Lambda_i & 0\\ 0 & 0\end{smallmatrix}\right]T^{\top}. \tag{2.27}$$

Since $Q^{1/2} \succ 0$ if $i = n$, then (2.27) admits $\Lambda = \Lambda_i$. Writing $T =: \begin{bmatrix}T_1 & T_2\end{bmatrix}$ implies

$$T^{\top}T = I \tag{2.28}$$

$$\begin{bmatrix}T_1 & T_2\end{bmatrix}\begin{bmatrix}T_1^{\top}\\ T_2^{\top}\end{bmatrix} = I \tag{2.29}$$

$$T_1T_1^{\top} + T_2T_2^{\top} = I \tag{2.30}$$

$$TT^{\top} = I \tag{2.31}$$

$$\begin{bmatrix}T_1^{\top}\\ T_2^{\top}\end{bmatrix}\begin{bmatrix}T_1 & T_2\end{bmatrix} = I \tag{2.32}$$

$$\begin{bmatrix}T_1^{\top}T_1 & T_1^{\top}T_2\\ T_2^{\top}T_1 & T_2^{\top}T_2\end{bmatrix} = \begin{bmatrix}I & 0\\ 0 & I\end{bmatrix} \tag{2.33}$$

Then, by selecting

$$\Upsilon = A_d^{1/2}(Z_{AB}-\zeta)Q^{-1/2} = A_d^{1/2}(Z_{AB}-\zeta)T_1\Lambda_i^{-1}T_1^\top, \tag{2.34}$$

it can be verified that $||\Upsilon|| \leq 1$ by applying some of the properties above.

$$\begin{aligned}
&\Upsilon^\top\Upsilon = T_1\Lambda_i^{-1}T_1^\top(Z-\zeta)^\top A_d^{1/2}A_d^{1/2}(Z-\zeta)T_1\Lambda_i^{-1}T_1^\top \\
\overset{(2.23)}{\Longrightarrow}\ &\Upsilon^\top\Upsilon \preceq T_1\Lambda_i^{-1}T_1^\top QT_1\Lambda_i^{-1}T_1^\top \\
\overset{(2.27)}{=}\ &T_1\Lambda_i^{-1}T_1^\top \begin{bmatrix} T_1 & T_2 \end{bmatrix} \begin{bmatrix} \Lambda_i & 0 \\ 0 & 0 \end{bmatrix} TT^\top \begin{bmatrix} \Lambda_i & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} T_1^\top \\ T_2^\top \end{bmatrix} T_1\Lambda_i^{-1}T_1^\top \\
\overset{(2.31)}{=}\ &T_1\Lambda_i^{-1}T_1^\top \begin{bmatrix} T_1 & T_2 \end{bmatrix} \begin{bmatrix} \Lambda_i^2 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} T_1^\top \\ T_2^\top \end{bmatrix} T_1\Lambda_i^{-1}T_1^\top \\
&= T_1\Lambda_i^{-1}T_1^\top T_1\Lambda_i^2T_1^\top T_1\Lambda_i^{-1}T_1^\top \overset{(2.33)}{=} T_1\Lambda_i^{-1}\Lambda_i^2\Lambda_i^{-1}T_1^\top \\
&= T_1T_1^\top = I
\end{aligned}$$

Then, (2.26) holds. Being (2.26) equivalent to

$$\Upsilon Q^{1/2} = \Upsilon \begin{bmatrix} T_1 & T_2 \end{bmatrix} \begin{bmatrix} \Lambda_i & 0 \\ 0 & 0 \end{bmatrix} T^\top = A_d^{1/2}(Z-\zeta) \tag{2.35}$$

$$\Longleftrightarrow \begin{bmatrix} \Upsilon T_1\Lambda_i & 0 \end{bmatrix} = A^{1/2}(Z-\zeta)T \tag{2.36}$$

$$\Longleftrightarrow \begin{bmatrix} \Upsilon T_1\Lambda_i & 0 \end{bmatrix} = A^{1/2}(Z-\zeta)\begin{bmatrix} T_1 & T_2 \end{bmatrix} \tag{2.37}$$

$$\Longleftrightarrow \begin{cases} \Upsilon T_1\Lambda_i = A^{1/2}(Z-\zeta)T_1 \\ 0 = A^{1/2}(Z-\zeta)T_2 \end{cases} \tag{2.38}$$

The upper equality of (2.38) holds because of the chosen Υ in (2.34). The lower equality of (2.38) holds if T_2 collumns are in $\ker(Q^{1/2})$ and $\ker(Q^{1/2}) \subseteq \ker(A_d^{1/2}(Z-\zeta))$. The first condition $-$ $T_2 \in \ker(Q^{1/2})$ $-$ is true because

$$Q^{1/2}T_2 = T \begin{bmatrix} \Lambda_i & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} T_1^\top \\ T_2^\top \end{bmatrix} T_2 \overset{(2.33)}{=} T \begin{bmatrix} \Lambda_i & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ I \end{bmatrix} = 0. \tag{2.39}$$

The second condition $-$ $\ker(Q^{1/2}) \subseteq \ker(A_d^{1/2}(Z_{AB}-\zeta))$ $-$ is also true, because if

$v \in \ker(Q^{1/2})$, v satisfies $Q^{1/2}v = 0$, thus, from (2.23),

$$\begin{aligned} 0 &= v^\top Q^{1/2} Q^{1/2} v \\ &= v^\top Q v \\ &\geq v^\top (Z-\zeta) A_d (Z-\zeta) v \\ &= |A_d^{1/2}(Z-\zeta)v|^2 \end{aligned}$$

Then, since $|A_d^{1/2}(Z-\zeta)v|^2 \leq 0$ and the absolute value of any matrix M is non-negative, i.e., $|M| \geq 0,\ \forall M$, it follows that $A_d^{1/2}(Z-\zeta)v = 0$. □

2.5.3 Petersen's lemma and control design

With the definition of set $\mathscr{C}$ as equivalent to set $\mathscr{E}$ (2.25), its representation of the uncertainty set that contains all pair of matrices $[A\ \ B]$ compatible with the data can be utilized in the process of obtaining some data-driven Lyapunov condition for control law design. For that, Petersen's lemma will now be presented, since it will be used in a key step further in this work.

Lemma 3 (Nonstrict Petersen's Lemma)**.** *Consider matrices $C \in \mathbb{R}^{n\times n}$, $E \in \mathbb{R}^{n\times p}$, $\bar{F} \in \mathbb{R}^{q\times q}$, $G \in \mathbb{R}^{q\times n}$, with $C = C^\top$ and $F = F^\top \succeq 0$, and let $\mathscr{F}$ be*

$$\mathscr{F} := \{F \in \mathbb{R}^{p\times q} : F^\top F \preceq \bar{F}\}. \tag{2.40}$$

Suppose additionally $E \neq 0$, $\bar{F} \succ 0$ and $G \neq 0$. Then,

$$C + EFG + G^\top F^\top E^\top \preceq 0, \quad \forall F \in \mathscr{F} \tag{2.41}$$

if and only if there exists $\lambda > 0$ such that

$$C + \lambda EE^\top + \lambda^{-1} G^\top \bar{F} G \preceq 0. \tag{2.42}$$

Proof. See (BISOFFI *et al.*, 2022) □

Bearing Petersen's lemma (Lemma 3), strict QSR-dissipativity (2.12), (SILVA *et al.*, 2024) devised a data-driven Lyapunov condition which, if satisfied, guarantees asymptotic stability of a system with a SOF control law.

Proposition 2. *Let data be given by Z_0, W_0 and X_1 as in (2.18). Assuming $\left[\begin{smallmatrix} Z_0 \\ W_0 \end{smallmatrix}\right]$ have full row rank and $Z(0) = 0$. Given positive scalars β_V, β_T, and positive integers n_V, n_T, if there*

exists polynomials $V(x)$, $T(x)$, $\lambda(x)$, *with* $V(0)=0$, $T(0)=0$, *and constant matrices* $Q\in\mathbb{S}^m$, $S\in\mathbb{R}^{p\times m}$, $R\in\mathbb{S}^m_{>0}$ *such that for each* x

$$V(x)-\beta_V||x||^{2n_V}\geq 0, \tag{2.43}$$

$$T(x)-\beta_T||x||^{2n_T}\geq 0, \tag{2.44}$$

$$\begin{bmatrix} \Sigma_1+\lambda(x)\Sigma_2 & \begin{bmatrix}\nabla V^T Q_d^{1/2}\\ 0\end{bmatrix} \\ \begin{bmatrix}Q_d^{1/2}\nabla V & 0\end{bmatrix} & -\lambda(x)I \end{bmatrix}\preceq 0, \tag{2.45}$$

$$\lambda(x)>0, \tag{2.46}$$

$$\Delta_c\succeq 0, \tag{2.47}$$

being

$$\Sigma_1=\begin{bmatrix}\nabla V^T\zeta_N^T Z(x)+T(x)-h(x)^T Qh(x) & \star\\ W(x)^T\zeta_M\frac{1}{2}\nabla V-S^T h(x) & -R\end{bmatrix}, \tag{2.48a}$$

$$\Sigma_2=\begin{bmatrix}\frac{1}{2}Z(x)^T(A_d^{-1/2})_N\\ \frac{1}{2}W(x)^T(A_d^{-1/2})_M\end{bmatrix}\cdot\begin{bmatrix}\star\end{bmatrix}, \tag{2.48b}$$

then the origin is globally asymptotically stable for $\dot{x}=AZ(x)+BW(x)u:=f_{A,B}(x)$, *for all* $\begin{bmatrix}A & B\end{bmatrix}\in\mathscr{C}$, *and a control law* $u=Kh(x)$, *with* $K=-R^{-1}S^T$.

Proof. For the proof, we start following the same rationale as (BISOFFI *et al.*, 2022) with the addition of the supply rate of the Dissipativity Theory (section 2.4). It will be shown that V is a Lyapunov function for all $\dot{x}=f_{A,B}(x)$, $\begin{bmatrix}A & B\end{bmatrix}\in\mathscr{C}$. From (2.43), V is radially unbounded, and since $V(0)=0$, it follows that V is positive definite. The next step is to address the derivative of V satisfying $\dot{V}=\nabla V^T f_{A,B}<0$. From (2.12), $\dot{V}<0$ if $\Delta_c\geq 0$ and $T(x)>0$. Then, the parametrization of $\begin{bmatrix}A & B\end{bmatrix}$ will be substituted in (2.23) to show that

$$\nabla V^T(AZ(x)+BW(x)u)\leq -T(x)+r(u,y(x))$$

if (2.45) and (2.46) both hold. Notice that the above inequality leads to

$$\nabla V^T\begin{bmatrix}A & B\end{bmatrix}\begin{bmatrix}Z(x)\\ W(x)u\end{bmatrix}+T(x)\leq h(x)^T Qh(x)+2h(x)^T Su+u^T Ru \tag{2.49}$$

$$\nabla V^T(\zeta+A_d^{-1/2}\Upsilon Q_d^{1/2})^T\begin{bmatrix}Z(x)\\ W(x)u\end{bmatrix}+T(x)-h(x)^T Qh(x)-2h(x)^T Su-u^T Ru\leq 0$$

$$\nabla V^T\zeta^T\begin{bmatrix}Z(x)\\ W(x)u\end{bmatrix}+\nabla V^T Q_d^{1/2}\Upsilon^T(A_d^{-1/2})^T\begin{bmatrix}Z(x)\\ W(x)u\end{bmatrix}+T(x)-h(x)^T Qh(x)-2h(x)^T Su-u^T Ru\leq 0$$

$$\nabla V^T \zeta^T \begin{bmatrix} Z(x) \\ W(x)u \end{bmatrix} + \frac{1}{2}\nabla V^T Q_d^{1/2} \Upsilon^T (A_d^{-1/2})^T \begin{bmatrix} Z(x) \\ W(x)u \end{bmatrix} + \begin{bmatrix} Z(x) \\ W(x)u \end{bmatrix}^T A_d^{-1/2} \Upsilon Q_d^{1/2} \frac{1}{2}\nabla V + T(x) - h(x)^T Q h(x) - 2h(x)^T S u - u^T R u \leq 0 \tag{2.50}$$

At this point, we differ from (BISOFFI *et al.*, 2022), by splitting ζ and $A_d^{-1/2}$ into its N-rows and M-rows components, $\zeta := \begin{bmatrix} \zeta_N \\ \zeta_M \end{bmatrix}$ and $A_d^{-1/2} := \begin{bmatrix} (A_d^{-1/2})_N \\ (A_d^{-1/2})_M \end{bmatrix}$. Then, we have

$$\begin{aligned} &\nabla V^T \zeta_N^T Z(x) + \nabla V^T \zeta_M^T W(x)u + \frac{1}{2}\nabla V^T Q_d^{1/2} \Upsilon^T (A_d^{-1/2})_N^T Z(x) \\ &+ \frac{1}{2}\nabla V^T Q_d^{1/2} \Upsilon^T (A_d^{-1/2})_M^T W(x)u + Z(x)^T (A_d^{-1/2})_N \Upsilon Q_d^{1/2} \frac{1}{2}\nabla V \\ &+ u^T W(x)^T (A_d^{-1/2})_M \Upsilon Q_d^{1/2} \frac{1}{2}\nabla V + T(x) - h(x)^T Q h(x) - 2h(x)^T S u - u^T R u \leq 0 \end{aligned} \tag{2.51}$$

Then, the expression above can be rearranged in the form of the matricial product bellow

$$\begin{bmatrix} 1 & u^T \end{bmatrix} . \begin{bmatrix} \nabla V^T (\zeta_N^T + Q_d^{1/2} \Upsilon^T (A_d^{-1/2})_N^T) Z(x) + T(x) - h(x)^T Q h(x) & \star \\ W(x)^T \left(\zeta_M + (A_d^{-1/2})_M \Upsilon Q_d^{1/2} \right) \frac{1}{2}\nabla V - S^T h(x) & -R \end{bmatrix} . \begin{bmatrix} 1 \\ u \end{bmatrix} \leq 0 \tag{2.52}$$

which is valid if the center matrix is negative semidefinite. Rewriting this matrix as a sum of two matrices, we obtain:

$$\begin{bmatrix} \nabla V^T \zeta_N^T Z(x) + T(x) - h(x)^T Q h(x) & \star \\ W(x)^T \zeta_M \frac{1}{2}\nabla V - S^T h(x) & -R \end{bmatrix} + \begin{bmatrix} \nabla V^T Q_d^{1/2} \Upsilon^T (A_d^{-1/2})_N^T Z(x) & \star \\ W(x)^T (A_d^{-1/2})_M \Upsilon Q_d^{1/2} \frac{1}{2}\nabla V & 0 \end{bmatrix} \preceq 0 \tag{2.53}$$

Then, we split the second matrix in as sum of matricial products in order to apply the nonstrict Petersen's lemma in the next steps

$$\begin{bmatrix} \nabla V^T \zeta_N^T Z(x) + T(x) - h(x)^T Q h(x) & \star \\ W(x)^T \zeta_M \frac{1}{2}\nabla V - S^T h(x) & -R \end{bmatrix} + \left([\star] + \begin{bmatrix} Z(x)^T (A_d^{-1/2})_N \Upsilon Q_d^{1/2} \frac{1}{2}\nabla V & 0 \\ W(x)^T (A_d^{-1/2})_M \Upsilon Q_d^{1/2} \frac{1}{2}\nabla V & 0 \end{bmatrix} \right) \preceq 0$$

$$\begin{bmatrix} \nabla V^T \zeta_N^T Z(x) + T(x) - h(x)^T Q h(x) & \star \\ W(x)^T \zeta_M \frac{1}{2}\nabla V - S^T h(x) & -R \end{bmatrix} + \left([\star] + \begin{bmatrix} \frac{1}{2} Z(x)^T (A_d^{-1/2})_N \\ \frac{1}{2} W(x)^T (A_d^{-1/2})_M \end{bmatrix} \Upsilon \begin{bmatrix} Q_d^{1/2} \nabla V & 0 \end{bmatrix} \right) \preceq 0 \tag{2.54}$$

By applying Petersen's lemma (Lemma 3), since $\Upsilon \Upsilon^T \leq I$, it's possible to obtain

$$\begin{bmatrix} \nabla V^T \zeta_N^T Z(x) + T(x) - h(x)^T Q h(x) & \star \\ W(x)^T \zeta_M \frac{1}{2}\nabla V - S^T h(x) & -R \end{bmatrix} + \lambda(x) \begin{bmatrix} \frac{1}{2} Z(x)^T (A_d^{-1/2})_N \\ \frac{1}{2} W(x)^T (A_d^{-1/2})_M \end{bmatrix} \begin{bmatrix} \star \end{bmatrix} + \lambda(x)^{-1} \begin{bmatrix} \nabla V^T Q_d^{1/2} \\ 0 \end{bmatrix} \begin{bmatrix} \star \end{bmatrix} \preceq 0 \tag{2.55}$$

Now, for concision, the matrices Σ_1 and Σ_2 are as (3.21), resulting in

$$\Sigma_1 + \lambda(x)\Sigma_2 - \begin{bmatrix} \nabla V^T Q_d^{1/2} \\ 0 \end{bmatrix} (-\lambda(x)^{-1}) \begin{bmatrix} Q_d^{1/2} \nabla V & 0 \end{bmatrix} \preceq 0$$

Finally, taking the Schur's complement of the inequation above leaves us with

$$\begin{bmatrix} \Sigma_1 + \lambda(x)\Sigma_2 & \begin{bmatrix} \nabla V^T Q_d^{1/2} \\ 0 \end{bmatrix} \\ \begin{bmatrix} Q_d^{1/2} \nabla V & 0 \end{bmatrix} & -\lambda(x) I \end{bmatrix} \preceq 0. \tag{2.56}$$

□

The condition stated in Proposition 2 can be applied as well when trying to achieve a local asymptotical stability. For that, an s-procedure is applied to (2.45), resulting in:

$$\begin{bmatrix} \Sigma_1 + \lambda(x)\Sigma_2 & \begin{bmatrix} \nabla V^T Q_d^{1/2} \\ 0 \end{bmatrix} \\ \begin{bmatrix} Q_d^{1/2} \nabla V & 0 \end{bmatrix} & -\lambda(x) I \end{bmatrix} + \alpha(x)(1 - L(x)) \preceq 0, \tag{2.57}$$

with a positive definite $\alpha(x) \in \mathscr{P}^{(n+m+1)\times(n+m+1)}$. The meaning of this condition is that if (2.57) is satisfied, (2.45) also holds in a region around the origin that is given by an ellipsoid $\mathscr{E}(L,1) := \{x \in \mathbb{R}^n \mid L(x) \leq 1\}$. This strategy can be useful when achieving global stability proves challenging or a limited domain of attraction is sufficient, which is true in many applications.

Alternatively to the supply rate in (2.10), it can be defined a more general supply rate as in (MADEIRA *et al.*, 2025) taking into account Q and S as polynomial matrices, i.e.,

$$r(u,y) = Q(y) + 2S(y)u + u^\top R u \tag{2.58}$$

Note that, for the purpose of this work, it is considered a real matrix R, since it allows for an simpler calculation of its inverse than if its considered polynomial. Then, a condition based in the supply rate in (2.58), similar to the one in Proposition 2 is formalized below.

Proposition 3. *Let data be given by Z_0, W_0 and X_1 as in (2.18). Assuming $\left[\begin{smallmatrix} Z_0 \\ W_0 \end{smallmatrix}\right]$ have full row rank and $Z(0) = 0$. Given positive scalars β_V, β_T, and positive integers n_V, n_T, if there exists polynomials $V(x)$, $T(x)$, $\lambda(x)$, with $V(0) = 0$, $T(0) = 0$, and $Q(y) \in \mathscr{P}$, $S(y) \in \mathscr{P}^{1\times m}$,*

$R(x) \in \mathbb{R}^{m\times m}_{>0}$ polynomial matrices such that for each xg

$$V(x) - \beta_V ||x||^{2n_V} \geq 0, \tag{2.59}$$

$$T(x) - \beta_T ||x||^{2n_T} \geq 0, \tag{2.60}$$

$$\begin{bmatrix} \Sigma_1 + \lambda(x)\Sigma_2 & \begin{bmatrix} \nabla V^T Q_d^{1/2} \\ 0 \end{bmatrix} \\ \begin{bmatrix} Q_d^{1/2}\nabla V & 0 \end{bmatrix} & -\lambda(x) I \end{bmatrix} \preceq 0, \tag{2.61}$$

$$\lambda(x) > 0, \tag{2.62}$$

$$\Delta_c \succeq 0, \tag{2.63}$$

being

$$\Sigma_1 = \begin{bmatrix} \nabla V^T \zeta_N^T Z(x) + T(x) - Q(y) & \star \\ W(x)^T \zeta_M \frac{1}{2}\nabla V - S(y) & -R \end{bmatrix}, \tag{2.64a}$$

$$\Sigma_2 = \begin{bmatrix} \frac{1}{2} Z(x)^T (A_d^{-1/2})_N \\ \frac{1}{2} W(x)^T (A_d^{-1/2})_M \end{bmatrix} \cdot \begin{bmatrix} \star \end{bmatrix}, \tag{2.64b}$$

then the origin is globally asymptotically stable for $\dot{x} = f_{A,B}(x)$, for all $\begin{bmatrix} A & B \end{bmatrix} \in \mathscr{C}$, and a control law $u = K(y)$, with $K = -R^{-1}S^{\top}(y)$.

Proof. The proof follows the exact same steps of the Proposition 2. □

The s-procedure can be applied as well in the Proposition 3 to achieve local asymptotical stability.

About the differences between Proposition 2 and Proposition 3, the first is based on a supply rate with constant matrices, which can lead to less computational effort when determining them through mathematical solvers. However, the supply rate in the second case (2.58) can offer a bigger set of possible solutions. Also, the definition of an output as in (2.5) is not needed and the degrees of complexity of the polynomial control law is defined by the $S(y)$ matrix. Furthermore, as seen in (MADEIRA *et al.*, 2025), the second proposition can be applied to obtain rational control laws, expanding their range of complexity.

3 METHODOLOGY

The initial step to design a data-driven control law is to perform an experiment for the data acquisition.

3.1 Collecting Data

For this work, the input signal selected was sinusoidal as in (BISOFFI *et al.*, 2022), which was applied in the systems state space model via simulation. Then, a number of samples T_s were extracted and the input and state sequences were used to construct the data-matrices Z_0, W_0 as in (2.18), with the noise added to the estimation of X_1 as in (2.20). A sampling time can be defined as τ_s, but here the simulation was made with MATLAB function `ode45`, and it utilizes a varying time step. This was not an issue since it did not affect the dataset's ability to describe the open-loop trajectories of the system.

The disturbance d was also modeled by sinusoidal waves. If its amplitude is given by $\sqrt{\frac{\delta}{n}}$ and frequency $2\pi\omega$, it makes $|d|^2 \leq \frac{\delta}{n}$, which in turn make the sequence D in (2.19) to be $D \preceq \sqrt{\frac{\delta}{n}} J_{n\times T_s}$, meaning that

$$DD^\top \preceq T_s \frac{\delta}{n} J_{n\times n}. \tag{3.1}$$

Now, analyzing the $J_{n\times n}$ matrix, it can be stated that $J_{n\times n} \preceq nI$. Such property can be derived from a particular case of the Cauchy-Schwarz inequality, i.e., $|\langle v, w\rangle|^2 \leq \langle v, v\rangle\langle w, w\rangle$, for any $v,\ w \in \mathbb{R}^n$, by selecting $w = J_{n\times 1}$, resulting in

$$\left(\sum_{i=1}^{n} v_i 1\right)^2 = \left(\sum_{i=1}^{n} v_i\right)^2 \leq \sum_{i=1}^{n} v_i^2 \sum_{i=1}^{n} 1^2 = n\sum_{i=1}^{n} v_i^2$$

Take into consideration that the left sum can be written in the matricial form $(\sum_{i=1}^{n} v_i)^2 = v^\top J_{n\times n} v$, and the sum in the left term can be written as $\sum_{i=1}^{n} v_i^2 = v^\top I v$, for any $v \in \mathbb{R}^n$, then $J_{n\times n} \preceq nI$. Then, $(3.1) \preceq T_s \frac{\delta}{n} nI$. This guarantees that $DD^\top \preceq T_s \delta I$, admitting $R_D = \sqrt{T_s \delta} I$.

Since the conditions for control law design in Proposition 2 and 3 are polynomial matrices, SOS Programming will be used to solve algorithms through SOS decomposition of polynomials.

3.2 SOS approach

A polynomial $p(x) \in \mathbb{R}^n$ can be written as a SOS decomposition if there is a positive semidefinite matrix M, such that

$$p(x) = z.M.z^\top, \quad z \in \mathbb{R}^m, \quad M \in \mathbb{R}^{m \times m} \tag{3.2}$$

where z contains the monomials of x in $p(x)$ and M contains the coeficients of such monomials. The utility of such decomposition is that it can be used to verify if a polynomial is positive semidefinite by finding its SOS representation (PAPACHRISTODOULOU *et al.*, 2021).

With SOSTools, a MATLAB toolbox, it can be defined a SOSP to find the SOS decomposition of polynomials, or polynomial matrices. Additionally in the SOSP, it can be defined decision variables, which are variables to be determined by the toolbox when finding the SOS decomposition, and some constraints for these same variables. The decision variables would be mainly the Lyapunov function $V(x)$ and the QSR-dissipativity matrices, along with the auxiliary variables shown in Proposition 2 and Proposition 3. Also, these decision variables are constrained by positive definite and semidefinite polynomial conditions, making this kind of toolbox fit perfectly in the problem studied in subsection 2.5.1. The Theorems 2 and 3 rewrite the mentioned propositions in the form of SOSP taking into account the s-procedure in (2.57) for local asymptotic stability.

Theorem 2. *Let data be given by Z_0, W_0 and X_1 as in (2.18). Assuming $\left[\begin{smallmatrix} Z_0 \\ W_0 \end{smallmatrix}\right]$ have full row rank and $Z(0) = 0$. Given positive scalars β_V, β_T, β_L, ε_λ, and positive integers n_V, n_T, n_L, if there exists polynomials $V(x)$, $T(x)$, $L(x)$,$\lambda(x)$, with $V(0) = 0$, $T(0) = 0$, constant matrices $Q \in \mathbb{S}^p$,*

$S \in \mathbb{R}^{p\times m}$, $R \in \mathbb{S}^m_{>0}$, and polynomial matrix $\alpha(x) \in \mathscr{P}^{(m+n+1)\times(m+n+1)}$ such that for each x

$$V(x) - \beta_V ||x||^{2n_V} \in \Sigma[x], \tag{3.3}$$

$$T(x) - \beta_T ||x||^{2n_T} \in \Sigma[x], \tag{3.4}$$

$$L(x) - \beta_L ||x||^{2n_L} \in \Sigma[x], \tag{3.5}$$

$$-\begin{bmatrix} \Sigma_1 + \lambda(x)\Sigma_2 & \begin{bmatrix} \nabla V^\top Q_d^{1/2} \\ 0 \end{bmatrix} \\ \begin{bmatrix} Q_d^{1/2}\nabla V & 0 \end{bmatrix} & -\lambda(x) I \end{bmatrix} - \alpha(x)(1-L(x)) \in \Sigma[x], \tag{3.6}$$

$$\lambda(x) - \varepsilon_\lambda \in \Sigma[x], \tag{3.7}$$

$$\alpha(x) \in \Sigma[x], \tag{3.8}$$

$$\Delta_c \in \Sigma[x], \tag{3.9}$$

being

$$\Sigma_1 = \begin{bmatrix} \nabla V^\top \zeta_N^\top Z(x) + T(x) - h(x)^\top Q h(x) & \star \\ W(x)^\top \zeta_M \frac{1}{2}\nabla V - S^\top h(x) & -R \end{bmatrix}, \tag{3.10a}$$

$$\Sigma_2 = \begin{bmatrix} \frac{1}{2} Z(x)^\top (A_d^{-1/2})_N \\ \frac{1}{2} W(x)^\top (A_d^{-1/2})_M \end{bmatrix} \cdot \begin{bmatrix} \star \end{bmatrix}, \tag{3.10b}$$

then the origin is locally asymptotically stable for $\dot{x} = f_{A,B}(x)$, for all $\begin{bmatrix} A & B \end{bmatrix} \in \mathscr{C}$, and a control law $u = Kh(x)$, with $K = -R^{-1}S^\top$. The domain of attraction is estimated by $\mathscr{E}(L,1)$.

Proof. Since a polynomial matrix having a SOS decomposition means it is positive semidefinite, (3.3), (3.4), (3.7), (3.9) imply respectively (2.43), (2.44), (2.46), (2.16). Additionally, (3.6) means that:

$$-\begin{bmatrix} \Sigma_1 + \lambda(x)\Sigma_2 & \begin{bmatrix} \nabla V^\top Q_d^{1/2} \\ 0 \end{bmatrix} \\ \begin{bmatrix} Q_d^{1/2}\nabla V & 0 \end{bmatrix} & -\lambda(x) I \end{bmatrix} - \alpha(x)(1-L(x)) \succeq 0. \tag{3.11}$$

Then, we can rewrite it as:

$$\begin{bmatrix} \Sigma_1 + \lambda(x)\Sigma_2 & \begin{bmatrix} \nabla V^\top Q_d^{1/2} \\ 0 \end{bmatrix} \\ \begin{bmatrix} Q_d^{1/2}\nabla V & 0 \end{bmatrix} & -\lambda(x) I \end{bmatrix} \preceq -\alpha(x)(1-L(x)). \tag{3.12}$$

Since $\alpha(x) \succeq 0$ because of (3.8), and $L(x) \geq 0$ because of (3.5), the right hand of the inequality above is only negative when $L(x) \leq 1$, i.e.,

$$\begin{bmatrix} \Sigma_1 + \lambda(x)\Sigma_2 & \begin{bmatrix} \nabla V^\top Q_d^{1/2} \\ 0 \end{bmatrix} \\ \begin{bmatrix} Q_d^{1/2}\nabla V & 0 \end{bmatrix} & -\lambda(x) I \end{bmatrix} \preceq -\alpha(x)(1-L(x)) \preceq 0, \quad x \in \mathscr{E}(L,1). \tag{3.13}$$

This means that, if (3.6) holds, (2.45) holds for all $x \in \mathscr{E}(L,1)$, being $\mathscr{E}(L,1)$ an estimative of the domain of attraction where the closed-loop system is dissipative. □

Theorem 3. *Let data be given by Z_0, W_0 and X_1 as in (2.18). Assuming $\left[\begin{smallmatrix} Z_0 \\ W_0 \end{smallmatrix}\right]$ have full row rank and $Z(0) = 0$. Given positive scalars β_V, β_T, β_L, ε_λ, and positive integers n_V, n_T, n_L, if there exists polynomials $V(x)$, $T(x)$, $\lambda(x)$, with $V(0) = 0$, $T(0) = 0$, and matrices $Q(y) \in \mathscr{P}$, $S(y) \in \mathscr{P}^{1\times m}$, $R \in \mathbb{S}^m_{>0}$, $\alpha(x) \in \mathscr{P}^{(n+m+1)\times(n+m+1)}$ such that for each x*

$$V(x) - \beta_V ||x||^{2n_V} \in \Sigma[x], \tag{3.14}$$

$$T(x) - \beta_T ||x||^{2n_T} \in \Sigma[x], \tag{3.15}$$

$$L(x) - \beta_L ||x||^{2n_L} \in \Sigma[x], \tag{3.16}$$

$$-\begin{bmatrix} \Sigma_1 + \lambda(x)\Sigma_2 & \begin{bmatrix} \nabla V^\top Q_d^{1/2} \\ 0 \end{bmatrix} \\ \begin{bmatrix} Q_d^{1/2}\nabla V & 0 \end{bmatrix} & -\lambda(x) I \end{bmatrix} - \alpha(x)(1-L(x)) \in \Sigma[x], \tag{3.17}$$

$$\lambda(x) - \varepsilon_\lambda \in \Sigma[x], \tag{3.18}$$

$$\alpha(x) \in \Sigma[x], \tag{3.19}$$

$$\Delta_c \in \Sigma[x], \tag{3.20}$$

being

$$\Sigma_1 = \begin{bmatrix} \nabla V^\top \zeta_N^\top Z(x) + T(x) - Q(y) & \star \\ W(x)^\top \zeta_M \frac{1}{2}\nabla V - S(y) & -R \end{bmatrix}, \tag{3.21a}$$

$$\Sigma_2 = \begin{bmatrix} \frac{1}{2} Z(x)^\top (A_d^{-1/2})_N \\ \frac{1}{2} W(x)^\top (A_d^{-1/2})_M \end{bmatrix} \cdot \begin{bmatrix} \star \end{bmatrix}, \tag{3.21b}$$

then the origin is locally asymptotically stable for $\dot{x} = f_{A,B}(x)$, for all $\begin{bmatrix} A & B \end{bmatrix} \in \mathscr{C}$, and a control law $u = K(y)$, with $K = -R^{-1}S^\top(y)$. The domain of attraction is estimated by $\mathscr{E}(L,1)$.

Proof. Following the same steps for the proof of Theorem 3, (3.14), (3.15), (3.18), (3.20) imply respectively (2.59), (2.44), (2.62), (2.63). Additionally, (3.17), (3.16), (3.19) mean that (2.61) holds for all $x \in \mathscr{E}(L,1)$, being $\mathscr{E}(L,1)$ an estimative of the domain of attraction where the closed-loop system is dissipative. □

In the previous theorems, $Z(x)$ and $W(x)$ are the known regressors of the plant, ζ, A_d and Q_d are matrices determined by the data Z_0, W_0 and X_1. The decision variables to be determined by the SOSP are $V(x)$, $T(x)$, $L(x)$, $\alpha(x)$, $\lambda(x)$, Q, S, R (or $Q(y)$, $S(y)$, R in Theorem 3). The input parameters β_V, β_T, β_L, n_V, n_T, n_L have influence in the radially unboundness of $V(x)$, $T(x)$ and $L(x)$. Since SOSTools does not compute contraints that are strict inequalities, the last input parameter $\varepsilon_\lambda > 0$ is needed to guarantee that $\lambda(x) > 0$, which is selected as a sufficient small value.

3.3 Algorithm Formulation

Both Theorems 2 and 3 present bilinearities that need to be addressed in order to be implemented. The first one is present the Δ_c as in (2.16). Since the problem needs to be linear in the decision variables, adding constraint in the form of (3.9)/(3.20) is not possible. This issue is circumvented in (MADEIRA; MACHADO, 2024) by two steps: the first step solves the SOSP with no restrictions in Δ_c, the second step is an iterative loop that solves the same SOSP with the addiction of a constraint that guarantees

$$\Delta_{c,k} \succeq \Delta_{c,k-1} \tag{3.22}$$

and stops when $\Delta_{c,\kappa} \geq 0$. Such condition is presented in the next proposition for the polynomial matrix case, based in (MADEIRA *et al.*, 2025), and can be applied in the real matrix case as well by making $Q(y) = Q$ and $S(y) = S$.

Proposition 4. *Suppose that* $\forall k = \{0,\ 1,\ 2\ , ...\}$*, there exist sequencies* $Q_k \in \mathscr{P}$*,* $S_k \in \mathscr{P}^{1\times m}$ *and* $R_k \in \mathbb{S}^m_{>0}$*. In addition, let* Δ_k *in (2.16) be well defined in x. If* $\forall k = \{0,\ 1,\ 2\ , ...\}$

$$R_{k-1} - R_k \succeq 0 \tag{3.23}$$

and

$$\begin{aligned} &S_k(y)R_{k-1}^{-1}S_{k-1}^{\top}(y) + S_{k-1}(y)R_{k-1}^{-1}S_k^{\top}(y) \\ &-2S_{k-1}(y)R_{k-1}^{-1}S_{k-1}^{\top}(y) + Q_{k-1}(y) - Q_k(y) \in \Sigma[y], \end{aligned} \tag{3.24}$$

then (3.22) holds $\forall k = \{0,\ 1,\ 2\ , ...\}$.

Proof. From (2.16), (3.22) can be written as

$$S_k(y)R_k^{-1}S_k^\top(y) - Q_k(y) \geq S_{k-1}(y)R_{k-1}^{-1}S_{k-1}^\top(y) - Q_{k-1}(y). \tag{3.25}$$

Suppose that (3.23) holds, then

$$(3.25) \impliedby S_k(y)R_{k-1}^{-1}S_k^\top(y) - S_{k-1}(x)R_{k-1}^{-1}S_{k-1}^\top(x) \geq Q_k(y) - Q_{k-1}(y). \tag{3.26}$$

Now, taking into account that

$$(S_k(y) - S_{k-1}(y))R_{k-1}^{-1}(S_k(y) - S_{k-1}(y))^\top \geq 0, \tag{3.27}$$

note it is equivalent to

$$S_k(y)R_{k-1}^{-1}S_k(y)^\top \geq S_k(y)R_{k-1}^{-1}S_{k-1}(y) + S_{k-1}(y)R_{k-1}^{-1}S_k(y) - S_{k-1}(y)R_{k-1}^{-1}S_{k-1}(y), \tag{3.28}$$

Then, by substitutig $S_k(x)R_{k-1}^{-1}S_k(x)^\top$ in (3.26), (3.26) is implied by

$$S_k(y)R_{k-1}^{-1}S_{k-1}(y) + S_{k-1}(y)R_{k-1}^{-1}S_k(y) - 2S_{k-1}(y)R_{k-1}^{-1}S_{k-1}(y) \geq Q_k(y) - Q_{k-1}(y), \tag{3.29}$$

which is identical to (3.24). Therefore (3.24) is a sufficient condition for (3.22). □

The importance of Proposition 4 is due to it providing a condition that is linear in the decision variables, i.e., $Q_k(y)$, $S_k(y)$ and R_k. Such conditions were named in (MADEIRA; MACHADO, 2024) as Recurrent Dissipativity-Based Inequalities (RDBI).

The second bilinearity present in Theorems 2 and 3 is in the s-procedure term $\alpha(x)(1-L(x))$. The method proposed in this work deals with it by, given some initial $\alpha(x)$, determining $L(x)$ in the first step. Then, in the iterative step, which utilize the RDBI, the $L(x)$ found in the first step is fixed, and only $\alpha(x)$ is treated as a decision variable. This method will be organized in the Algorithms 1 and 2 presented next.

In Algorithm 1, since $\Delta_c \in \mathbb{R}$, the test $\Delta_c \succeq 0$ required in the `if` sections can be performed by analyzing if all of its eigenvalues are positive. In Algorithm 2, such test cannot be as easily performed. Since $\Delta_c \in \mathscr{P}$, an alternative test can be made by an auxiliary SOSP to find if $\Delta_c \in \Sigma[x]$, thus satisfying $\Delta_c \geq 0$.

Algorithm 1: Data-driven Local Dissipativity Control Design - supply rate as in (2.10)

Input : Consider a $Z(x) \in \mathscr{P}^{N\times 1}$, $W(x) \in \mathscr{P}^{M\times 1}$, $h(x) \in \mathscr{P}^{p\times 1}[x]$, set $(\beta_V, \beta_T, \beta_L, \varepsilon_\lambda) \in \mathbb{R}_{>0}$, $(n_V, n_T, n_L, k_{max}) \in \mathbb{N}$, $\alpha_0(x) \in \mathscr{P}_{>0}^{(n+m+1)\times(n+m+1)}$

$k \leftarrow 0$

STEP 1) Find $Q_0 \in \mathbb{S}^p$, $S_0 \in \mathbb{R}^{p\times m}$, $R_0 \in \mathbb{S}_{>0}^m$, $\lambda_0(x) \in \mathscr{P}$, $V_0(x) \in \mathscr{P}$, $T_0(x) \in \mathscr{P}$, $L(x) \in \mathscr{P}$

subject to (3.3)-(3.7)

if $\Delta_{c,0} \succeq 0$ **then**
  $K = -R_0^{-1} S_0^T$
  $V = V_0(x)$
else
  while $k < k_{max}$ **do**
    STEP 2) Find $Q_k \in \mathbb{S}^p$, $S_k \in \mathbb{R}^{p\times m}$, $R_k \in \mathbb{S}_{>0}^m$, $\lambda_k(x) \in \mathscr{P}$, $V_k(x) \in \mathscr{P}$, $T_k(x) \in \mathscr{P}$, $\alpha(x) \in \mathscr{P}^{(n+m+1)\times(n+m+1)}$

    subject to (3.3)-(3.7), (3.23)-(3.24) and $\alpha(x) \in \Sigma^{(n+m+1)\times(n+m+1)}[x]$

    if $\Delta_{c,k} \geq 0$ ***or*** $k = k_{max}$ **then**
      $K = -R_k^{-1} S_k^T$
      $V = V_k(x)$
      STOP
    end
    $k \leftarrow k+1$
  end
end

Output : K, $V(x)$

3.4 Lyapunov Certificate with Z3 Prover

Bearing the solutions found by Algorithms 1 and 2, a Lyapunov certificate can be obtained by verifying if Z3 Prover can find a counterexample for $V(x) > 0$ and $\dot{V} < 0$ in the estimated domain of attraction $\mathscr{E}(L,1)$. This feature is illustrated below with an code example for Z3 Prover in Python which attempts to prove if $-x^2 + 10 < 0$ when $x < 5$, $x \in \mathbb{R}$. Code:

```
import z3
x= z3.Real('x')
z3.prove(z3.Implies(x<=5,-x**2+10>=0))
```

Output:

```
counterexample
[x = 4]
```

Algorithm 2: Data-driven Local Dissipativity Control Design - supply rate as in (2.58)

Input : Consider a $Z(x) \in \mathscr{P}^{N\times 1}$, $W(x) \in \mathscr{P}^{M\times 1}$, set $(\beta_V, \beta_T, \beta_L, \varepsilon_\lambda) \in \mathbb{R}_{>0}$, $(n_V, n_T, n_L, k_{max}) \in \mathbb{N}$, $\alpha_0(x) \in \mathscr{P}_{>0}^{(n+m+1)\times(n+m+1)}$

$k \leftarrow 0$

STEP 1) Find $Q_0(y) \in \mathscr{P}$, $S_0(y) \in \mathscr{P}^{1\times m}$, $R_0 \in \mathbb{S}_{>0}^m$, $\lambda_0(x) \in \mathscr{P}$, $V_0(x) \in \mathscr{P}$, $T_0(x) \in \mathscr{P}$, $L(x) \in \mathscr{P}$

subject to (3.14)-(3.18)

if $\Delta_{c,0} \in \Sigma[y]$ **then**
  $K = -R_0^{-1} S_0^T(y)$
  $V = V_0(x)$
else
  while $k < k_{max}$ **do**
    STEP 2) Find $Q_k(y) \in \mathscr{P}$, $S_k(y) \in \mathscr{P}^{1\times m}$, $R_k \in \mathbb{S}_{>0}^m$, $\lambda_k(x) \in \mathscr{P}$, $V_k(x) \in \mathscr{P}$, $T_k(x) \in \mathscr{P}$, $\alpha(x) \in \mathscr{P}^{(n+m+1)\times(n+m+1)}$

    subject to (3.14)-(3.18), (3.23)-(3.24) and $\alpha(x) \in \Sigma^{(n+m+1)\times(n+m+1)}[x]$

    if $\Delta_{c,k} \in \Sigma[x]$ ***or*** $k = k_{max}$ **then**
      $K = -R_k^{-1} S_k^T(y)$
      $V = V_k(x)$
      STOP
    end
    $k \leftarrow k+1$
  end
end

Output : $K(y)$, $V(x)$

Alternatively, if not found a counterexample for the formula, it is considered proved or satisfied as the next example illustrates. Code:

```
import z3
x= z3.Real('x')
z3.prove(z3.Implies(x>=4,x**2-10>=0))
```

Output:

```
proved
```

Because they're able to prove the positive definiteness of a polynomial within a domain, SMT solvers can be used to provide a *Lyapunov certificate* by checking if the data-driven Lyapunov condition in (2.45)/(2.61) holds with the solutions found by Algorithm 1/2. Moreover, there is a scenario where the positive definite constraints can be feasible, but a SOS

decomposition could not be found, e.g., due to numerical imprecision or due to the polynomial not having an SOS decomposition. This scenario may cause the SOSP solvers to return solutions not certifiable via SOSP, but SMT solvers can still provide the Lyapunov certificate.

The procedure for issuing a Luapunov certicate was to verify if the Lyapunov conditions (Theorem 1) were satisfied. For $V(x) > 0$, it can be a straight-forward test by verifying its positive-definiteness as illustrated in example codes above. For $\dot{V}(x) < 0$, the tested condition was the data-driven conditions 2.45/2.61, but with a Schur Complement in order to make it a scalar polynomial (instead of a matrix polynomial), since it cannot be analyzed by Z3 Prover otherwise.

In addition to that, its important to remark that the estimative of the domain of attraction $\mathscr{E}(L,1)$ is only determined in STEP 1 for both algorithms. If all steps were feasible, Lyapunov conditions can be satisfied in an equal or bigger domain of attraction with the final solutions found. In this work, with the solutions provided by SOSTools, Z3 Prover was applied to provide a Lyapunov certificate and to maximize the domain of attraction by conducting a line search in ρ to find a maximum $\mathscr{E}(V,\rho)$ in which (2.45)/(2.61) holds.

4 RESULTS

To apply the strategy developed in the previous section, two systems with a nonlinear polynomial state-space representation were selected. The local asymptotic stabilization around the origin of both systems was solved using both algorithms 1 and 2, for comparison in the use of the different definitions of the supply rate ((2.10) and (2.58)). The software used to implement the algorithms was MATLAB, particularly with the SOSTools toolbox, using Mosek as solver. The solutions found were then tested with Z3 Theorem Prover to guarantee a Lyapunov certificate. The computer in which the software was running has the specifications: Processor 12th Gen Intel®Core ™i7-12700F 2.10 GHz, with RAM 32,0 GB.

4.1 System 1

The first system was extracted from (KHALIL, 2002, Example 14.9), and it can be represented by the following state-space equations:

$$\begin{cases} \dot{x}_1 = x_1^2 - x_1^3 + x_2 \\ \dot{x}_2 = u \end{cases} \tag{4.1}$$

Figure 3 – Block diagram of system (4.1)

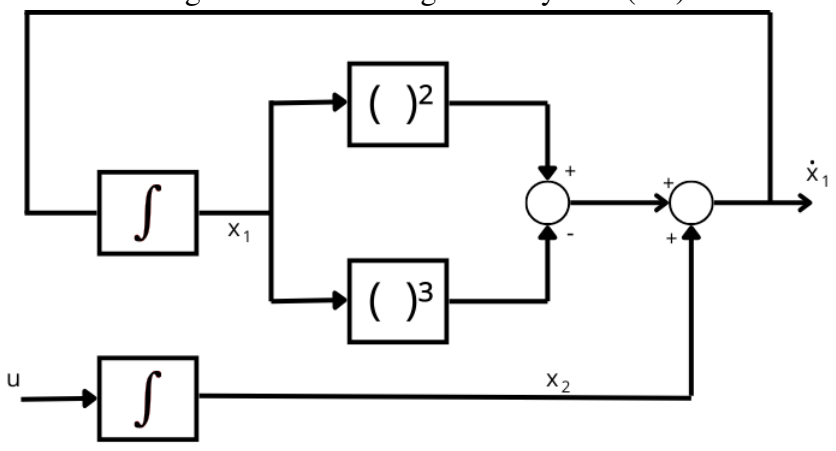
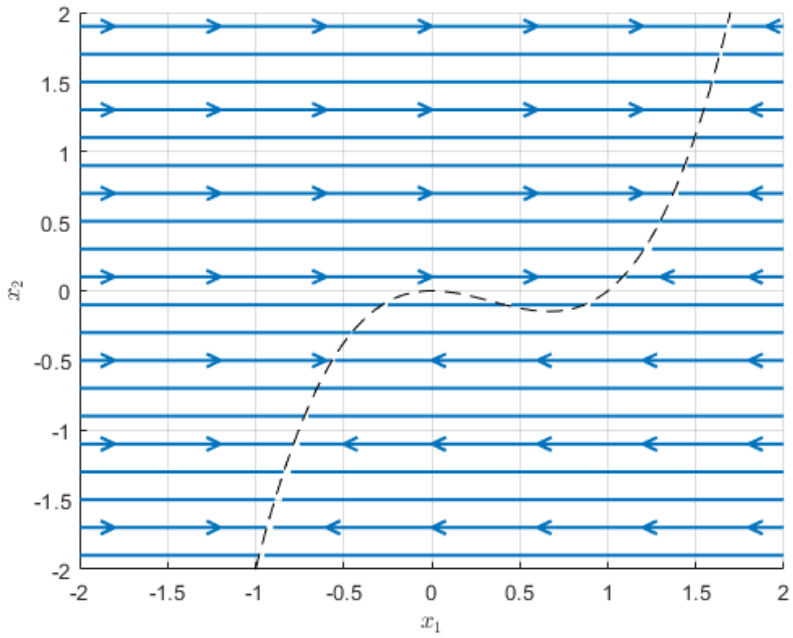


Figure 3 shows the block diagram for system (4.1) and the open-loop phase diagram is shown in figure 5. The system already converges for a set of equilibrium points, when $\dot{x}_1 = x_1^2 - x_1^3 + x_2 = 0$, shown as a dashed line. However, we want to achieve asymptotical stability around the origin (as a isolated equilibrium point).

Figure 4 – Open-loop phase diagram for system (4.1)

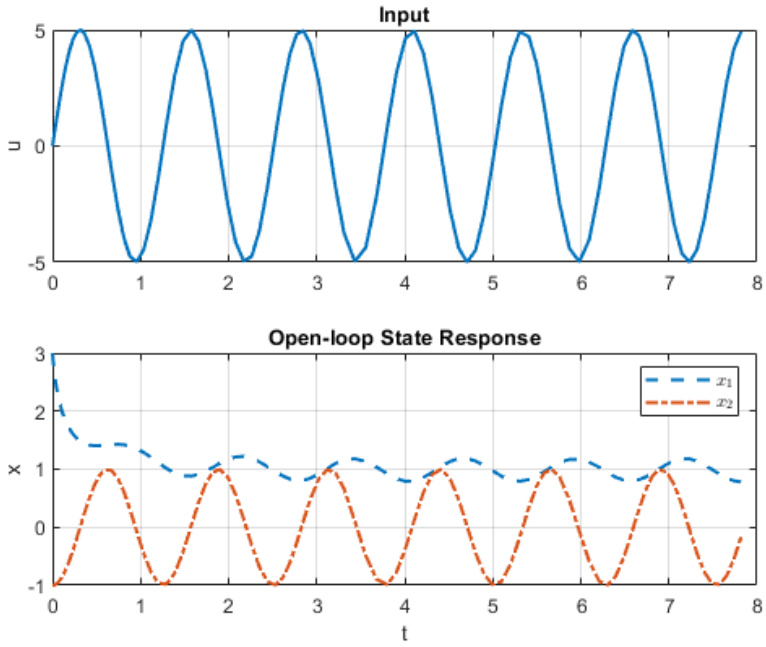


To perform the data acquisition experiment, the input signal selected was $u = 5\sin 5t$, in the interval $t \in [0, 10]$, t in seconds, and it was extracted the first $T_s = 100$ samples provided by MATLAB's `ode45` function. That particular signal was chosen considering the main assumption in Lemma 1, i.e., $\left[\begin{smallmatrix} Z_0 \\ W_0 \end{smallmatrix}\right]$ having full row rank, with selected vectors of monomials being $Z = [x_1\ x_1^2\ x_1^3\ x_2\ x_2^2\ x_2^3]$ and $W = [1]$. The disturbance signal applied in the sequence X_1 was $d = \left[\begin{smallmatrix} \sqrt{\delta/n}\sin\omega t \\ \sqrt{\delta/n}\cos\omega t \end{smallmatrix}\right]$, with $\delta = 0.01$ and $\omega = 0.8\pi$, which satisfies (2.19). This disturbance emulates the process disturbances as modelled in section 3.1, in order to design a control law that robustly stabilize all pair all matrices $[A\ \ B]$ consistent with data. The experiment data is shown in figure 5.

4.1.1 Algorithm 1

For algorithm 1, the input parameters were $n_V = n_T = n_L = 1$, $\beta_V = \beta_T = 10^{-6}$, $\beta_L = 10^{-4}$, $\varepsilon_\lambda = 10^{-3}$, and $\alpha_0(x) = 10^{-6}||x||^2 I$. The fictitious output, which determines the polynomial degrees of the control law obtained, was $h(x) = [x_1\ x_2]^\top$. The degrees of the polynomial decision variables $V(x)$, $T(x)$, $\lambda(x)$, $L(x)$, $\alpha(x)$ were respectively 4, 2, 0, 2 and 2. By selecting this inputs, the computational complexity of algorithm 1 can be assessed by the number of decision variables and constraints in the SOS programs executed in STEP 1 and STEP 2. In STEP 1, it were 18 decision variables and 6 SOS constraints, and in STEP 2, 45 decision variables and 8 SOS constraints. The execution time until a solution with $\Delta_c \succeq 0$ was 5.953207

Figure 5 – Open-loop experiment for system (4.1)



seconds and the number of iterations was $k = 37$. Table 1 shows the solutions found by algorithm 1 for system (4.1).

Table 1 – $V(x)$, $u(x)$, $\lambda(x)$ and $L(x)$ solutions for (4.1), supply rate as in (2.10)

Fcn	Expression
V	$0.077217x_1^4 + 0.058274x_1^3x_2 + 0.27619x_1^2x_2^2 - 0.11577x_1x_2^3 + 0.015567x_2^4$
u	$-7.834x_1 - 5.6198x_2$
λ	0.0072974
L	$200977679.8299x_1^2 - 220011.7166x_1x_2 + 215571661.4608x_2^2$

With Z3Prover, it was possible to obtain a Lyapunov certificate by not finding a counterexample for the Lyapunov conditions (Theorem 1) for asymptotic stability in the domain of attraction estimated by $\mathscr{E}(L,1)$. In addition, by conducting a line search in ρ to find a bigger ellipsoid $\mathscr{E}(L,\rho)$, it was obtained a Lyapunov certificate in a domain of attraction with $\rho = 10^8$.

At the left side of Figure 6, it is shown the phase diagram for the closed-loop, with the dashed line representing the estimated domain of attraction $\mathscr{E}(L,10^8)$ and the bold line representing a single trajectory for $x_0 = [0.4\ 0.4]$, which is also represented in the time domain at the right side of Figure 6.

4.1.2 Algorithm 2

For algorithm 2, the input parameters were $n_V = n_T = n_L = 1$, $\beta_V = \beta_T = 10^{-6}$, $\beta_L = 10^{-4}$, $\varepsilon_\lambda = 10^{-3}$, and $\alpha_0(x) = 10^{-6}||x||^2 I$. With the new supply, the complexity of the control law is determined by $S(x)$, since $u = -R^{-1}S^\top(x)$. Then, the degrees of the polynomial decision variables $V(x)$, $T(x)$, $\lambda(x)$, $L(x)$, $\alpha(x)$, $Q(x)$, $S(x)$ were respectively 4, 2, 0, 2, 2, 2 and 1. Assessing the computational complexity, STEP 1 had 18 decision variables and 6 SOS and STEP 2, 45 decision variables and 8 SOS constraints. The execution time until a solution with $\Delta_c \succeq 0$ was 6.601079 seconds and the number of iterations was $k = 37$. Table 2 shows the solutions found by algorithm 2 for system (4.1).

Table 2 – $V(x)$, $u(x)$, $\lambda(x)$ and $L(x)$ solutions for (4.1), supply rate as in (2.58)

Fcn	Expression
V	$0.06835x_1^4 + 0.061552x_1^3x_2 + 0.27479x_1^2x_2^2 - 0.11572x_1x_2^3 + 0.016102x_2^4$
u	$-8.1515x_1 - 5.2592x_2$
λ	0.0075004
L	$165169341.5667x_1^2 + 1112927.3042x_1x_2 + 190941194.2397x_2^2$

Similar to the previous example, it was possible to obtain a Lyapunov certificate with Z3 and the maximum domain of attraction $\mathscr{E}(L,\rho)$ found was for also $\rho = 10^8$.

At the left side of Figure 7, it is shown the phase diagram for the closed-loop, with the dashed line representing the estimated domain of attraction $\mathscr{E}(L,10^8)$ and the bold line representing a single trajectory for $x_0 = [0.4\ 0.4]$, which is also represented in the time domain at the right side of Figure 7.

4.1.3 Comparison with (BISOFFI et al., 2022)

In the same way as in (SILVA *et al.*, 2024), the results above were compared to the method for data-driven control for polynomial systems in (BISOFFI *et al.*, 2022), but in here we address the (BISOFFI *et al.*, 2022, Corollary 3) for local asymptotical stability. The data was produced with the same experiment as in Figure 5.

The input parameters and decision variables were selected trying to keep them in similar complexity as their equivalents in the two previous subsections, but still aiming for

successful stabilization. Then, the selected parameters were $l_1 = 10^{-6}||x||^2$ (equivalent to β_V and n_V), $\varepsilon_\lambda = 10^{-8}$, initial $V(x) = x^\top \left[\begin{smallmatrix} 0.0278 & 0.0127 \\ 0.0127 & 0.0216 \end{smallmatrix}\right] x$ (obtained by stabilizing the linear model), $l_0 = x^\top x$, $c = 1$ (similar to $L(x)$, but here the domain of attraction $\mathscr{E}(l_0, 1)$is a input parameter), $k_{max} = 15$ (stop criterion). The degrees of the polynomial decision variables $V(x)$, $k(x)$ (control law), $\lambda(x)$, $T(x)$, $s_1(x)$, $s_2(x)$ (s-procedure variables similar to $\alpha(x)$ were respectively 2 to 4, 1, 0, 2 to 4, 2 to 4, 2 to 4. In terms of computational complexity, STEP 1 ($V(x)$ fixed) had 3 decision variables and 6 SOS and STEP 2 ($k(x)$ fixed), 12 decision variables and 9 SOS constraints. The execution time until $k = k_{max}$ was 142.638647 seconds. Table 3 shows the solutions found by (BISOFFI *et al.*, 2022, Corollary 3) for system (4.1).

Table 3 – $V(x)$, $u(x)$ and $\lambda(x)$ solutions by (BISOFFI *et al.*, 2022, Corollary 3)

Fcn	Expression
V	$-2.5176 \times 10^{-9} x_1^4 + 7.8577 \times 10^{-9} x_1^3 x_2 - 2.2778 \times 10^{-7} x_1^2 x_2^2 - 2.8085 \times 10^{-8} x_1 x_2^3$ $-2.3159 \times 10^{-7} x_2^4 + 1.1703 \times 10^{-7} x_1^3 + 4.5677 \times 10^{-8} x_1^2 x_2 + 3.8531 \times 10^{-8} x_1 x_2^2$ $+1.7155 \times 10^{-9} x_2^3 + 5.4677 \times 10^{-7} x_1^2 + 4.8187 \times 10^{-8} x_1 x_2 + 8.5266 \times 10^{-7} x_2^2$
u	$-1.0959 x_1 - 3.2629 x_2$
λ	12960469.1712

With Z3Prover, it was possible to obtain a Lyapunov certificate by not finding a counterexample for the Lyapunov conditions (Theorem 1) for asymptotic stability in the domain of attraction estimated by $\mathscr{E}(L, 1)$. In addition, by conducting a line search in ρ to find a bigger ellipsoid $\mathscr{E}(L, \rho)$, it was obtained a Lyapunov certificate in a domain of attraction with $\rho = 1$.

At the left side of Figure 8, it is shown the phase diagram for the closed-loop, with the dashed line representing the estimated domain of attraction $\mathscr{E}(L, 1)$ and the bold line representing a single trajectory for $x_0 = [0.4\ 0.4]$, which is also represented in the time domain at the right side of Figure 8.

Figure 6 – Closed-loop phase diagram (left) and time response (right) for system (4.1) using Algorithm 1

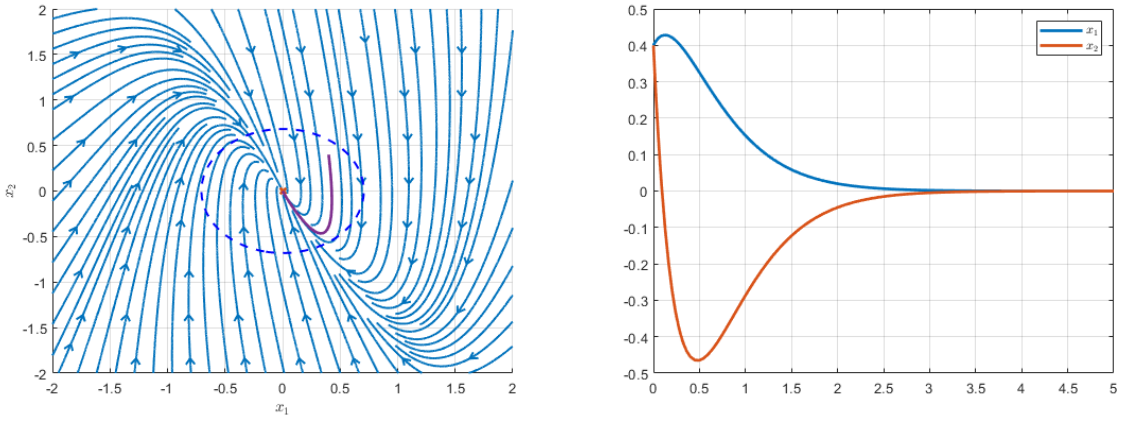


Figure 7 – Closed-loop phase diagram (left) and time response (right) for system (4.1) using Algorithm 2

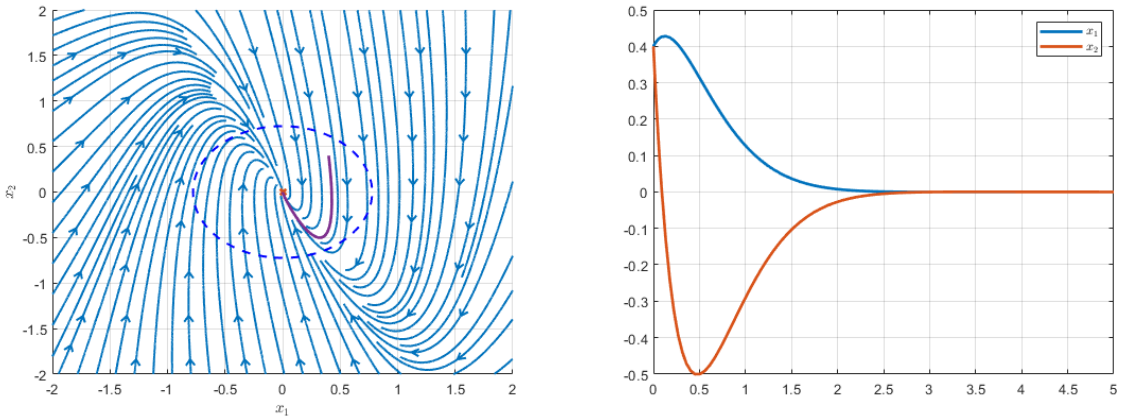
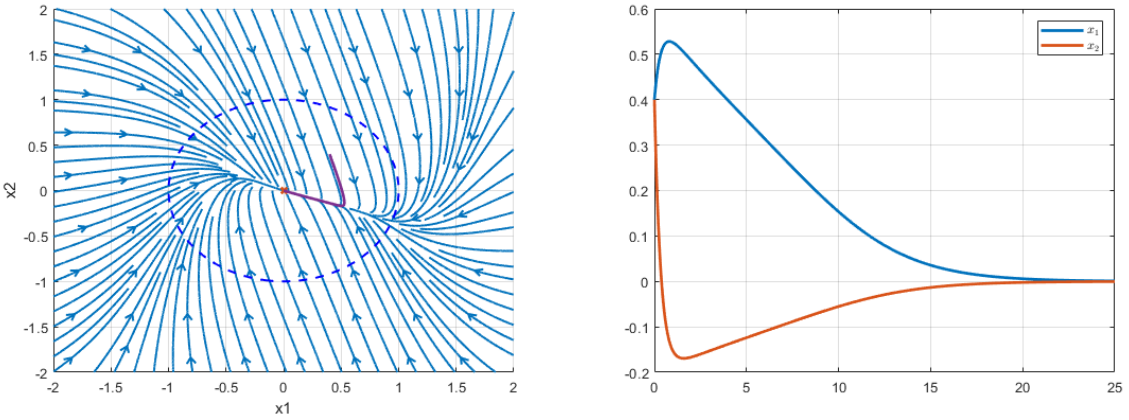


Figure 8 – Closed-loop phase diagram (left) and time response (right) for system (4.1) using Algorithm in (BISOFFI *et al.*, 2022)



By observing the closed-loop phase diagrams (Figures 6, 7, 8), the domain of attraction of the last method was bigger. However, by comparing the execution time and computational complexity, it had worse performance in total time and single iteration time,

despite having less decision variables in general constraints per STEP, even with different solvers. Table 7 shows the total execution time and total iterations of Algorithms 1, 2 and from (BISOFFI *et al.*, 2022) using Mosek (APS, 2025) and SeDuMi (STURM, 1999) as the solvers (SeDuMi results obtained from (SILVA *et al.*, 2024)). This could be due to the main SOS condition (the one derived from $\dot{V} < 0$ being bigger and/or it being more sparse than the ones present in the other two algorithms, causing numerical complexity to scale up. Moreover, the D-K iteration method have to solve 2 SOSP per iteration to deal with the bilinearity of the constraints, have no stop criterion beside the maximum number of iteration, and has no solution convergence constraint between iterations. Algorithms 1 and 2 tackle these issues by having only 1 SOSP per iteration, a stop criterion when $\Delta_c \succeq 0$ and a solution convergence constraint in $\Delta_{c,k+1} \succeq \Delta_{c,k}$ (3.22). Additionally, although all algorithms had no constraint to specify a time response (such maximum overhoot or settling time), the last algorithm resulted in the slowest closed-loop time response. In general, it successfully stabilized the system in the origin, but the performance of the algorithm based in (BISOFFI *et al.*, 2022, Corollary 3) was inferior to the Algorithms 1 and 2.

Table 4 – Algorithm performances for system (4.1)

Solver	Algorithm	Total execution time (s)	Total iterations
Mosek	1	5.953207	37
	2	6.601079	37
	(BISOFFI *et al.*, 2022)	142.638647	15
SeDuMi	1	15.07	4
	(BISOFFI *et al.*, 2022)	8023.18	15

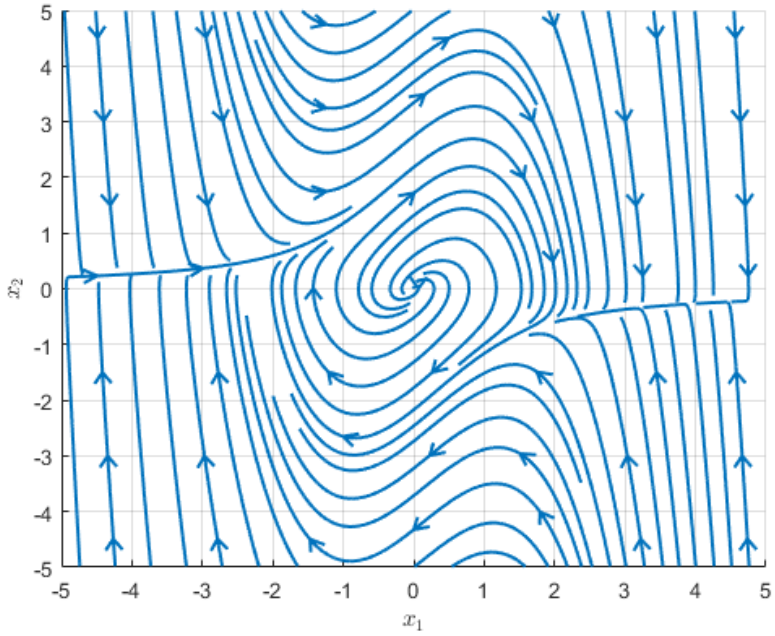
4.2 System 2 - Van der Pol's Oscillator

The second system was extracted from (HADDAD; CHELLABOINA, 2008). It is also known as Van der Pol's Oscillator and it can be represented by the following state-space equations:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -x_1 + e * (1 - x_1^2) * x_2 + u \end{cases} \tag{4.2}$$

The open-loop phase diagram is shown in figure 9, and the system is not asymptotically stable around the origin and converges to a periodic trajectory around the origin.

Figure 9 – Open-loop phase diagram for system (4.2)



To perform the data acquisition experiment, the input signal selected was $u = 5\sin 5t$, in the interval $t \in [0, 10]$, t in seconds, and it was extracted the first $T_s = 100$ samples provided by MATLAB's `ode45` function. That particular signal was chosen considering the main assumption in lemma 1, i.e., $\begin{bmatrix} Z_0 \\ W_0 \end{bmatrix}$ having full row rank, with selected vectors of monomials being Z all the monomials of x with degree from 1 to 3 and $W = [1]$. The disturbance signal applied in the sequence X_1 was $d = \begin{bmatrix} \sqrt{\delta/n}\sin \omega t \\ \sqrt{\delta/n}\cos \omega t \end{bmatrix}$, with $\delta = 0.01$ and $\omega = 0.8\pi$, the same used in section 4.2. The experiment data is shown in figure 10.

4.2.1 Algorithm 1

For algorithm 1, the input parameters were $n_V = n_T = n_L = 1$, $\beta_V = \beta_T = 10^{-6}$, $\beta_L = 10^{-4}$, $\varepsilon_\lambda = 10^{-8}$, and $\alpha_0(x) = 10^{-4}||x||^2 I$. The monomial degrees of the fictitious output $h(x)$ was $\{1, 3\}$. The degrees of the polynomial decision variables $V(x)$, $T(x)$, $\lambda(x)$, $L(x)$, $\alpha(x)$ were respectively 2 to 4, 2, 0, 2 to 4 and 2 to 6. These parameters resulted in 56 decision variables and 6 SOS constraints in STEP 1 of the algorithm, and 294 decision variables and 8 constraints in STEP 2. The execution time until a solution with $\Delta_c \succeq 0$ was 8.084261 seconds and the number of iterations was $k = 15$. Table 5 shows the solutions found by algorithm 1 for system (4.2).

It was possible to obtain a Lyapunov certificate with Z3 and the maximum domain of attraction $\mathscr{E}(L, \rho)$ found was for $\rho = 10$.

At the left side of Figure 11, it is shown the phase diagram for the closed-loop,

Figure 10 – Open-loop experiment for system (4.2)

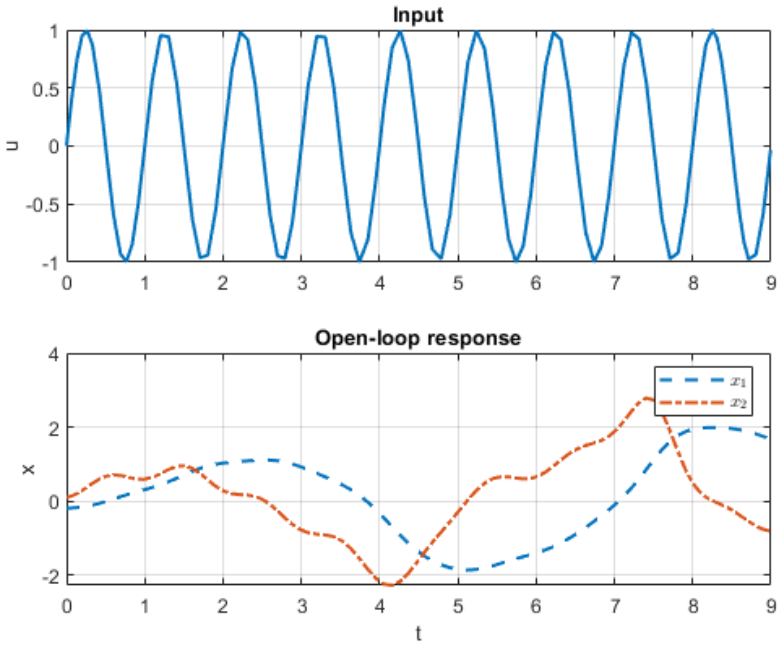


Table 5 – $V(x)$, $u(x)$, $\lambda(x)$ and $L(x)$ solutions for (4.2), supply rate as in (2.10)

Fcn	Expression
V	$0.214x_1^4 + 0.25868x_1^3x_2 + 0.46486x_1^2x_2^2 + 0.12299x_1x_2^3 + 0.70013x_2^4 + 0.0023443x_1^3$ $-0.0062013x_1^2x_2 + 0.0020025x_1x_2^2 - 0.00013813x_2^3 + 1.0024x_1^2 + 0.78862x_1x_2$ $+0.41352x_2^2$
u	$3.2224x_1^3 - 2.8409x_1^2x_2 + 3.6794x_1x_2^2 - 10.5933x_2^3 - 6.828x_1 - 8.9324x_2$
λ	3.9818
L	$10.9632x_1^4 + 0.0084373x_1^3x_2 + 12.5655x_1^2x_2^2 + 0.0054906x_1x_2^3 + 10.9904x_2^4$ $-1.3268 \times 10^{-5}x_1^3 - 7.921 \times 10^{-5}x_1^2x_2 - 1.4367 \times 10^{-5}x_1x_2^2 - 6.5266 \times 10^{-5}x_2^3$ $+0.78079x_1^2 - 3.6692 \times 10^{-5}x_1x_2 + 0.78044x_2^2$

with the dashed line representing the estimated domain of attraction $\mathscr{E}(L,10)$ and the bold line representing a single trajectory for $x_0 = [0.3\ 0.4]$, which is also represented in the time domain at the right side of Figure 11.

Figure 11 – Closed-loop phase diagram (left) and time response (right) for system (4.2) using Algorithm 1

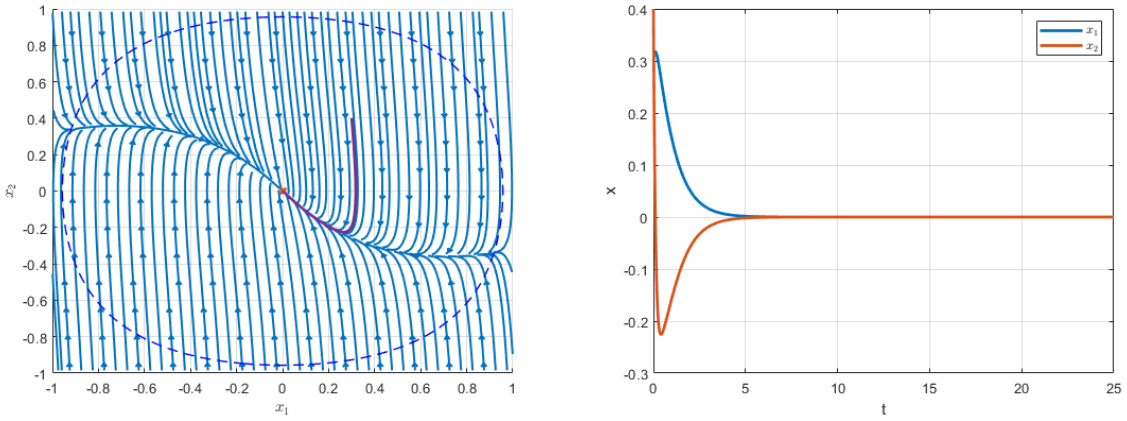
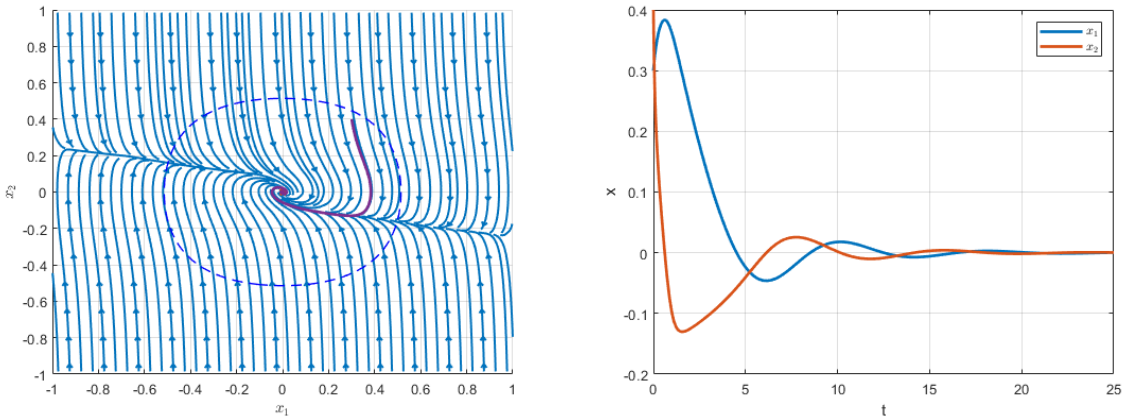


Figure 12 – Closed-loop phase diagram (left) and time response (right) for system (4.2) using Algorithm 2



4.2.2 Algorithm 2

For algorithm 2, the input parameters were $n_V = n_T = n_L = 1$, $\beta_V = \beta_T = 10^{-6}$, $\beta_L = 10^{-4}$, $\varepsilon_\lambda = 10^{-8}$, and $\alpha_0(x) = 10^{-4}||x||^2 I$. Then, the degrees of the polynomial decision variables $V(x)$, $T(x)$, $\lambda(x)$, $L(x)$, $\alpha(x)$, $Q(x)$, $S(x)$ were respectively 2 to 4, 2, 0, 2 to 4, 2 to 6, 2 to 6 and $\{1,3\}$. These parameters resulted in 60 decision variables and 6 SOS constraints in STEP 1 of the algorithm, and 298 decision variables and 8 constraints in STEP 2. The execution time until a solution with $\Delta_c \succeq 0$ was 5.729857 seconds and the number of iterations was $k = 6$. Table 5 shows the solutions found by algorithm 2 for system (4.2).

It was possible to obtain a Lyapunov certificate with Z3 and the maximum domain of attraction $\mathscr{E}(L,\rho)$ found was for $\rho = 1$.

At the left side of Figure 12, it is shown the phase diagram for the closed-loop, with the dashed line representing the estimated domain of attraction $\mathscr{E}(L,1)$ and the bold line

Table 6 – $V(x)$, $u(x)$, $\lambda(x)$ and $L(x)$ solutions for (4.2), supply rate as in (2.58)

Fcn	Expression
V	$7.201\times10^{-19}x_1^4 - 1.5867\times10^{-15}x_1^3x_2 + 5.1148\times10^{-15}x_1^2x_2^2 - 2.5178\times10^{-14}x_1x_2^3$ $+9.8052\times10^{-14}x_2^4 + 9.2091\times10^{-18}x_1^3 + 8.6472\times10^{-17}x_1^2x_2 + 3.2336\times10^{-16}x_1x_2^2$ $-1.044\times10^{-15}x_2^3 - 7.3986\times10^{-17}x_1^2 - 3.6131\times10^{-15}x_1x_2 - 8.7963\times10^{-15}x_2^2$
u	$-1.4412x_1^3 - 1.9801x_1^2x_2 + 8.7331x_1x_2^2 - 44.7184x_2^3 + 0.32905x_1 - 1.449x_2$
λ	-3.14×10^{-17}
L	$10.704x_1^4 + 0.0038086x_1^3x_2 + 11.8831x_1^2x_2^2 + 0.0035042x_1x_2^3 + 10.7213x_2^4$ $-0.00010667x_1^3 + 0.00011662x_1^2x_2 - 7.3657\times10^{-5}x_1x_2^2 + 0.00016152x_2^3$ $+0.94605x_1^2 + 4.0593\times10^{-5}x_1x_2 + 0.94607x_2^2$

representing a single trajectory for $x_0 = [0.3\ 0.4]$, which is also represented in the time domain at the right side of Figure 12. Table 7 shows the total execution time and total iterations of Algorithms 1, 2.

Table 7 – Algorithm performances for system (4.2)

Solver	Algorithm	Total execution time (s)	Total iterations
Mosek	1	8.084261	15
	2	5.729857	6

5 CONCLUSION AND FUTURE WORKS

In this work, data-driven QSR-dissipativity-based conditions for control design of polynomial systems were developed with two supply rates: with constant and polynomial Q, S, R matrices. In the implementation of these conditions, both algorithms succesfully designed a control law that locally asymptotically stabilized the two analyzed systems.

Comparing the performance of the algorithms, the use of a supply rate with polynomial Q, S, R matrices achieved a solution with faster total execution time and fewer iterations than the algorithm with constant Q, S, R matrices. Additionally, when comparing with another SOSP method (based on the D-K iteration), although the proposed algorithms had more decision variables and constraints, their execution time was smaller.

At last, the auxiliary use of Z3Prover in providing Lyapunov certificates for the found solutions a valuable source of reassuring the Lyapunov conditions are being satisfied, specially since numerical imprecisions can lead the softwares like SOSTools to not finding a SOS decomposition as well.

Future works can extend the approach developed here to achieve rational control laws and another formal verification tools can be explored, e.g., SMT solvers that can provide a neural Lyapunov certificate by training a neural network to act as Lyapunov function.

BIBLIOGRAPHY

ABATE, A.; AHMED, D.; GIACOBBE, M.; PERUFFO, A. Formal synthesis of lyapunov neural networks. **IEEE Control Systems Letters**, v. 5, n. 3, p. 773–778, 2021.

AHMED, D.; PERUFFO, A.; ABATE, A. Automated and sound synthesis of lyapunov functions with smt solvers. In: ____. [S.l.: s.n.], 2020. p. 97–114. ISBN 978-3-030-45189-9.

APS, M. **MOSEK Modeling Cookbook**. [S.l.], 2025. Disponível em: <https://docs.mosek.com/modeling-cookbook/index.html>.

AUGUST, E.; PAPACHRISTODOULOU, A. Feedback control design using sum of squares optimisation. **European Journal of Control**, v. 68, p. 100683, 2022. ISSN 0947-3580. 2022 European Control Conference Special Issue. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0947358022000759>.

BISOFFI, A.; PERSIS, C. D.; TESI, P. Data-driven control via petersen's lemma. **Automatica**, Elsevier BV, v. 145, p. 110537, nov. 2022. ISSN 0005-1098. Disponível em: <http://dx.doi.org/10.1016/j.automatica.2022.110537>.

HADDAD, W.; CHELLABOINA, V. **Nonlinear Dynamical Systems and Control: A Lyapunov-Based Approach**. [S.l.]: Princeton University Press, 2008. ISBN 9781400841042.

HORN, R.; JOHNSON, C. **Matrix Analysis**. Cambridge University Press, 2013. (Matrix Analysis). ISBN 9780521839402. Disponível em: <https://books.google.com.br/books?id=5I5AYeeh0JUC>.

ICHIHARA, H. A convex approach to state feedback synthesis for polynomial nonlinear systems with input saturation. **SICE Journal of Control, Measurement, and System Integration**, Taylor & Francis, v. 6, n. 3, p. 186–193, 2013.

KHALIL, H. **Nonlinear Systems**. Prentice Hall, 2002. (Pearson Education). ISBN 9780130673893. Disponível em: <https://books.google.com.br/books?id=t_d1QgAACAAJ>.

LÖFBERG, J. Yalmip : A toolbox for modeling and optimization in matlab. In: **In Proceedings of the CACSD Conference**. Taipei, Taiwan: [s.n.], 2004.

MADEIRA, D. de S.; MACHADO, G. F. Recurrent dissipativity-based inequalities for controller design. **IFAC-PapersOnLine**, v. 58, n. 21, p. 19–24, 2024. ISSN 2405-8963. 4th IFAC Conference on Modelling, Identification and Control of Nonlinear Systems MICNON 2024. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2405896324018913>.

MADEIRA, D. de S.; SILVA, J. G. N.; MACHADO, G.; PAPACHRISTODOULOU, A. Nonlinear static output feedback design for polynomial systems with non-symmetric input saturation bounds. **IEEE Control Systems Letters**, PP, p. 1–1, 01 2025.

MARTIN, T.; ALLGöWER, F. Data-driven system analysis of nonlinear systems using polynomial approximation. **IEEE Transactions on Automatic Control**, v. 69, n. 7, p. 4261–4274, 2024.

MOURA, L. de; BJøRNER, N. Z3: an efficient smt solver. In: . [S.l.: s.n.], 2008. v. 4963, p. 337–340. ISBN 978-3-540-78799-0.

PAPACHRISTODOULOU, A.; ANDERSON, J.; VALMORBIDA, G.; PRAJNA, S.; SEILER, P.; PARRILO, P.; PEET, M.; JAGT, D.; PAPACHRISTODOULOU, A.; ANDERSON, J.; VALMOR-BIDA, G.; PRAJNA, S.; PEET, M. Sum of squares optimization toolbox for matlab user's guide. 12 2021.

PERSIS, C.; TESI, P. Formulas for data-driven control: Stabilization, optimality, and robustness. **IEEE Transactions on Automatic Control**, PP, p. 1–1, 12 2019.

SILVA, J. G. N.; MADEIRA, D. d. S.; CORREIA, W. B. Data-driven control of polynomial systems using dissipativity-based inequalities and petersen's lemma. In: **2024 12th International Conference on Control, Mechatronics and Automation (ICCMA)**. [S.l.: s.n.], 2024. p. 89–94.

STURM, J. F. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. **Optimization methods and software**, Taylor & Francis, v. 11, n. 1-4, p. 625–653, 1999.

WILLEMS, J. C. Dissipative dynamical systems. **European Journal of Control**, v. 13, n. 2, p. 134–151, 2007. ISSN 0947-3580. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0947358007708166>.

WILLEMS, J. C.; RAPISARDA, P.; MARKOVSKY, I.; De Moor, B. L. A note on persistency of excitation. **Systems Control Letters**, v. 54, n. 4, p. 325–329, 2005. ISSN 0167-6911. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167691104001434>.

APPENDIX A – CODES

Code 1 – Control design algorithm - system 1 - supply rate with constant matrices Q, S, R

```
% % % Data-driven Control - Dissipativity-based with Petersen's Lemma
% % % Author: Joao Gabriel Napoleao Silva - NEACON - UFC
% % % System: Khalil (2008)

clc; clear; close all

% Data-driven Dissipativity Polynomial Control Synthesis
pvar x1 x2
vars = [x1;x2];
x = [x1;x2];

% f = [x1^2-x1^3+x2;
%       0];
%
% g = [0; 1];
%
h = monomials(x,1); %fictitious output

m = 1; % u columns
n = 2; % x columns
p = size(h,1); % h length

%------------------------------ DATA COLLECT ------------------------------%
global delta
delta = 1e-2;   % Disturbance amplitude

x0 = [3; -1]; % initial condition
```

```
opt = odeset('RelTol', 1e-3, 'AbsTol', 1e-6, 'NormControl', 'off');
[t, xsol] = ode45(@experiment, [0 10], x0, opt); % simulation time = [0,10]
X00 = xsol';
U00 = [];
X11 = [];

for i = 1:length(t)
    U00(:,i) = 5*sin(5*t(i)); % control signal
    X11(:,i) = [X00(1,i)^2-X00(1,i)^3+X00(2,i)+sqrt(delta/n)*cos(2*pi*0.4*t(i)); % derivative signal
                U00(:,i)+sqrt(delta/n)*sin(2*pi*0.4*t(i))];
end

Ts = 100; % number of samples

X0 = X00(:,1:Ts);    % selecting first T samples of the experiment
X1 = X11(:,1:Ts);
U0 = U00(:,1:Ts);
t = t(1:Ts);

DELTA = sqrt(Ts*delta)*eye(n); % Disturbance upper bound

% Initialization of regressors
Z = [monomials(x1,1:3);monomials(x2,1:3)];
W = monomials(x,0);

N = length(Z);
M = length(W);

```

```
% plot of generated data
figure
subplot(2,1,1)
input_exp = plot(t,U0);
title('Input');
set(input_exp,'LineWidth',1.5);
ylabel('u')
grid on

subplot(2,1,2)
states_exp = plot(t,X0(1,:),'--',t,X0(2,:),'-.');
legend('$x_1$','$x_2$','Interpreter','latex','Location','
   northeast');
title('Open-loop State Response');
set(states_exp,'LineWidth',1.5);
ylabel('x')
xlabel('t')
grid on

% construction of the data matrices Ad, Bd, Cd
Z0 = double(subs(Z,x,X0));
W0 = U0;

Ad = [Z0;W0]*[Z0;W0]';
Bd = -[Z0;W0]*X1';
Cd = X1*X1'-DELTA*DELTA';

% construction of the data matrices zeta and Q
zeta = -Ad\Bd;
Qd = Bd'*(Ad\Bd)-Cd;

zeta_N = zeta(1:N,:);
```

```
87  zeta_M = zeta((N+1):(N+M),:);
88  Ad_i = Ad^(-1/2);
89  Ad_i_N = Ad_i(:,1:N);
90  Ad_i_M = Ad_i(:,(N+1):(N+M));
91
92  % Inputs (design parameters)
93  beta_V = 1e-6;
94  beta_T = 1e-6;
95  beta_L = 1e-4;
96  n_V = 1;
97  n_T = 1;
98  n_L = 1;
99  epsi = 1e-3;
100 alpha = 1e-6*(x1^2+x2^2)*eye(n+m+1);
101
102 % Maximum iteration number
103 k = 1;
104 kmax = 100;
105
106 tic
107 % STEP 1: Determine V0, Tx, L0, Q0, S0, R0, lambda0
108 prog = sosprogram(vars);
109
110 [prog, Q] = sospolymatrixvar(prog,monomials(x,0),[p,p],'
       symmetric');
111 [prog, S] = sospolymatrixvar(prog,monomials(x,0),[p,m]);
112 [prog, R] = sospolymatrixvar(prog,monomials(x,0),[m,m],'
       symmetric');
113 [prog, V] = sospolyvar(prog,[monomials(x,4)],'wscoeff');
114 [prog, T] = sospolyvar(prog,[monomials(x,2)],'wscoeff');
115 [prog, lambda] = sospolyvar(prog,[monomials(x,0)],'wscoeff'
       );
```

```
116 [prog, L] = sospolyvar(prog,[monomials(x,2)],'wscoeff');
117
118 % SOS constrains
119 prog = sosineq(prog,V-beta_V*(x1^2+x2^2)^n_V);
120 prog = sosineq(prog,T-beta_T*(x1^2+x2^2)^n_T);
121 prog = sosineq(prog,L-beta_L*(x1^2+x2^2)^n_L);
122 gradV = jacobian(V,x)'; % gradient of Lyapunov function
123
124 Sigma1 = [gradV'*zeta_N'*Z+T-h'*Q'*h (1/2)*gradV'*zeta_M'*W
       -h'*S
125           ((1/2)*gradV'*zeta_M'*W-h'*S)' -R];
126 Sigma2 = [(1/2)*Z'*Ad_i_N'; (1/2)*W'*Ad_i_M']*[(1/2)*Ad_i_N
       *Z (1/2)*Ad_i_M*W];
127 B_aux = [gradV'*Qd^(1/2)
128         zeros(m,n)];
129
130 stability = [Sigma1+lambda*Sigma2 B_aux          % data-
       driven dissipativity-based condition for local stability
131                 B_aux' -lambda*eye(n)]+alpha*(1-L);
132
133 prog = sosineq(prog,-stability);
134
135 prog = sosineq(prog,R);      % dissipativity's R constraint
136
137 prog = sosineq(prog,lambda-epsi); % petersen's lemma's
       lambda constraint
138
139 % STEP 1 solution
140 options.solver = 'mosek';
141 sol = sossolve(prog,options);
142
143 Q0 = double(sosgetsol(sol,Q));
```

```
144 R0 = double(sosgetsol(sol,R));
145 S0 = double(sosgetsol(sol,S));
146 L = sosgetsol(sol, L);
147
148 T = sosgetsol(sol,T);
149 Delta_c = S0*(R0\(S0'))-Q0;
150 min(eig(Delta_c))
151
152 if min(eig(Delta_c)) >= 0   % Stop criteria: Delta_c test
153     K = -R0\(S0')           % Control gains
154     V = sosgetsol(sol,V)    % Lyapunov function
155 else
156     % STEP 2: Iterative method for V, Tx, alpha, Q, S, R,
            lambda
157     while k <= kmax
158         prog = sosprogram(vars);
159
160         [prog, Q] = sospolymatrixvar(prog,monomials(x,0),[p
                ,p],'symmetric');
161         [prog, S] = sospolymatrixvar(prog,monomials(x,0),[p
                ,m]);
162         [prog, R] = sospolymatrixvar(prog,monomials(x,0),[m
                ,m],'symmetric');
163         [prog, V] = sospolyvar(prog,[monomials(x,4)],'
                wscoeff');
164         [prog, T] = sospolyvar(prog,[monomials(x,2)],'
                wscoeff');
165         [prog, lambda] = sospolyvar(prog,[monomials(x,0)],'
                wscoeff');
166         [prog, alpha] = sospolymatrixvar(prog,[monomials(x
                ,2)],[n+m+1,n+m+1],'symmetric');
167
```

```
168         % SOS constraints
169         prog = sosineq(prog,V-beta_V*(x1^2+x2^2)^n_V);
170         prog = sosineq(prog,T-beta_T*(x1^2+x2^2)^n_T);
171
172         gradV = jacobian(V,x)';   % gradient of Lyapunov
               function
173
174         Sigma1 = [gradV'*zeta_N'*Z+T-h'*Q'*h (1/2)*gradV'*
               zeta_M'*W-h'*S
175                 ((1/2)*gradV'*zeta_M'*W-h'*S)' -R];
176         Sigma2 = [(1/2)*Z'*Ad_i_N'; (1/2)*W'*Ad_i_M
               ']*[(1/2)*Ad_i_N*Z (1/2)*Ad_i_M*W];
177         B_aux = [gradV'*Qd^(1/2)
178               zeros(m,n)];
179
180         stability = [Sigma1+lambda*Sigma2 B_aux
                        % data-driven dissipativity-based
               condition for local stability
181                       B_aux' -lambda*eye(n)]+alpha*(1-L);
182
183         prog = sosineq(prog,-stability);
184
185         prog = sosineq(prog,R);                % dissipativity
               's R constraint
186
187         prog = sosineq(prog,R0-R);             % increasing
               delta constraints
188         delta_increasing = S*(R0\(S0')) + (R0\S0')'*S' - 2*
               S0*(R0\(S0')) + Q0 - Q;
189         prog = sosineq(prog,delta_increasing);
190
```

```
191         prog = sosineq(prog,lambda-epsi); % petersen's
               lemma's lambda constraint
192         prog = sosineq(prog,alpha);
193
194         % STEP 2 solution
195         sol = sossolve(prog,options);
196
197         Q = double(sosgetsol(sol,Q))
198         R = double(sosgetsol(sol,R))
199         S = double(sosgetsol(sol,S))
200         alpha = sosgetsol(sol,alpha);
201         lambda = sosgetsol(sol,lambda);
202
203         T = sosgetsol(sol,T);
204         Delta_c = S*(R\(S'))-Q;
205         min(eig(Delta_c))
206         if min(eig(Delta_c))>= 0 | k == kmax     % Stop
               criteria: Delta_c test
207             K = -R\(S')                         % Control
                   gains
208             V = sosgetsol(sol,V)                % Lyapunov
                   function
209             break
210         end
211         k=k+1
212         Q0 = Q; R0 = R; S0 = S;
213     end
214 end
215 toc
216
217 % Phase diagram - Open loop
218 syms x1 x2
```

```
219 a=2;
220 [x1,x2] = meshgrid(-a:0.1:a,-a:0.1:a);
221
222 dx1 = x1.^2 - x1.^3 + x2;
223 dx2 = 0.*x1;
224
225 figure
226 phase_OL = streamslice(x1,x2,dx1,dx2,1);
227 set(phase_OL,'LineWidth',1.5);
228 title('Open-loop phase diagram')
229 xlabel('$x_1$','Interpreter','latex'), ylabel('$x_2$','
       Interpreter','latex')
230 grid on
231 %%
232 % Phase diagram - Closed loop
233 syms x1 x2
234 global Kx
235 Kx = matlabFunction(p2s(K*h));
236 a=4;
237 passo = 0.01;
238 [x1,x2] = meshgrid(-a:passo:a,-a:passo:a);
239
240 dx1 = x1.^2 - x1.^3 + x2;
241 dx2 = Kx(x1,x2);
242
243 figure
244 phase_CL = streamslice(x1,x2,dx1,dx2,1.5);
245 set(phase_CL,'LineWidth',1.5);
246 title('Closed-loop phase diagram','')
247 xlabel('$x_1$','Interpreter','latex'), ylabel('$x_2$','
       Interpreter','latex')
248 hold on
```

```
plot(0,0,'x','LineWidth',2)
grid on
elipL = matlabFunction(p2s(1-L));
zhandle = fimplicit(elipL);
zhandle.LineWidth = 1.5;
zhandle.LineStyle = "-";
zhandle.Color = "b";
x0 = [1; 2]; % initial condition
[t, xsol] = ode45(@experiment_CL, [0 15], x0, opt);
plot(xsol(:,1),xsol(:,2),'LineWidth',2);
figure
plot(t,xsol(:,1),t,xsol(:,2),'LineWidth',2);
title('Time response - closed-loop')

grid on
legend('$x_1$','$x_2$','Interpreter','latex','Location','
   northeast');

%------------------------------ EXPERIMENT
   -------------------------------%
function dx = experiment(t,x)
    global delta
    u = 5*sin(5*t);
    dx = [x(1)^2-x(1)^3+x(2);
          u];
end

function dx = experiment_CL(t,x)
    global Kx
    dx = [x(1)^2-x(1)^3+x(2);
          Kx(x(1),x(2))];
end
```

Code 2 – Control design algorithm - system 1 - supply rate with polynomial matrices Q, S, R

```
clc; clear; close all
%%% Data-driven Control - Dissipativity-based with Petersen's Lemma
%%% Author: Joao Gabriel Napoleao Silva - NEACON - UFC
%%% System: Khalil (2008)

% Data-driven Dissipativity Polynomial Control Synthesis
pvar x1 x2
vars = [x1;x2];
x = [x1;x2];

% f = [x1^2-x1^3+x2;
%       0];
%
% g = [0; 1];

m = 1; % u columns
n = 2; % x columns

%------------------------------ DATA COLLECT ------------------------------%

global delta
delta = 1e-2;   % Disturbance amplitude

x0 = [3; -1]; % initial condition
opt = odeset('RelTol', 1e-3, 'AbsTol', 1e-6, 'NormControl', 'off');
[t, xsol] = ode45(@experiment, [0 10], x0, opt); % simulation time = [0,10]
X00 = xsol';
```

```
28 U00 = [];
29 X11 = [];
30
31 for i = 1:length(t)
32     U00(:,i) = 5*sin(5*t(i)); % control signal
33     X11(:,i) = [X00(1,i)^2-X00(1,i)^3+X00(2,i)+sqrt(delta/n
       )*cos(2*pi*0.4*t(i)); % derivative signal
34                 U00(:,i)+sqrt(delta/n)*sin(2*pi*0.4*t(i))];
35 end
36
37 Ts = 100; % number of samples
38
39 X0 = X00(:,1:Ts);    % selecting first T samples of the
   experiment
40 X1 = X11(:,1:Ts);
41 U0 = U00(:,1:Ts);
42 t = t(1:Ts);
43
44 DELTA = sqrt(Ts*delta)*eye(n); %Disturbance upper bound
45
46 % Initialization of regressors
47 Z = [monomials(x1,1:3);monomials(x2,1:3)];
48 W = monomials(x,0);
49
50 N = length(Z);
51 M = length(W);
52
53 % plot of generated data
54 figure
55 subplot(2,1,1)
56 input_exp = plot(t,U0);
57 title('Input');
```

```
58 set(input_exp,'LineWidth',1.5);
59 ylabel('u')
60 grid on
61
62 subplot(2,1,2)
63 states_exp = plot(t,X0(1,:),'--',t,X0(2,:),'-.');
64 legend('$x_1$','$x_2$','Interpreter','latex','Location','
      northeast');
65 title('Open-loop State Response');
66 set(states_exp,'LineWidth',1.5);
67 ylabel('x')
68 xlabel('t')
69 grid on
70
71 % construction of the data matrices Ad, Bd, Cd
72 Z0 = double(subs(Z,x,X0));
73 W0 = U0;
74
75 Ad = [Z0;W0]*[Z0;W0]';
76 Bd = -[Z0;W0]*X1';
77 Cd = X1*X1'-DELTA*DELTA';
78
79 % construction of the data matrices zeta and Q
80 zeta = -Ad\Bd;
81 Qd = Bd'*(Ad\Bd)-Cd;
82
83 zeta_N = zeta(1:N,:);
84 zeta_M = zeta((N+1):(N+M),:);
85 Ad_i = Ad^(-1/2);
86 Ad_i_N = Ad_i(:,1:N);
87 Ad_i_M = Ad_i(:,(N+1):(N+M));
88
```

```
% Inputs (design parameters)
beta_V = 1e-6;
beta_T = 1e-6;
beta_L = 1e-4;
n_V = 1;
n_T = 1;
n_L = 1;
epsi = 1e-3;
alpha = 1e-6*(x1^2+x2^2)*eye(n+m+1);

% Maximum iteration number
k = 1;
kmax = 100;

tic
% STEP 1: Determine V0, Tx, L0, Q0, S0, R0, lambda0
prog = sosprogram(vars);

[prog, Q] = sospolyvar(prog,monomials(x,2),'wscoeff');
[prog, S] = sospolyvar(prog,monomials(x,1),'wscoeff');
[prog, R] = sospolymatrixvar(prog,monomials(x,0),[m,m],'
   symmetric');
[prog, V] = sospolyvar(prog,[monomials(x,4)],'wscoeff');
[prog, T] = sospolyvar(prog,[monomials(x,2)],'wscoeff');
[prog, lambda] = sospolyvar(prog,[monomials(x,0)],'wscoeff'
   );
[prog, L] = sospolyvar(prog,[monomials(x,2)],'wscoeff');

% SOS constrains
prog = sosineq(prog,V-beta_V*(x1^2+x2^2)^n_V);
prog = sosineq(prog,T-beta_T*(x1^2+x2^2)^n_T);
prog = sosineq(prog,L-beta_L*(x1^2+x2^2)^n_L);
```

```
119 gradV = [diff(V,x1); diff(V,x2)]; % gradient of Lyapunov
    function
120
121 Sigma1 = [gradV'*zeta_N'*Z+T-Q (1/2)*gradV'*zeta_M'*W-S
122           ((1/2)*gradV'*zeta_M'*W-S)' -R];
123 Sigma2 = [(1/2)*Z'*Ad_i_N'; (1/2)*W'*Ad_i_M']*[(1/2)*Ad_i_N
    *Z (1/2)*Ad_i_M*W];
124 B_aux = [gradV'*Qd^(1/2)
125        zeros(m,n)];
126
127 stability = [Sigma1+lambda*Sigma2 B_aux          % data-
    driven dissipativity-based condition for local stability
128                B_aux' -lambda*eye(n)]+alpha*(1-L);
129
130 prog = sosineq(prog,-stability);
131
132 prog = sosineq(prog,R);     % dissipativity's R constraint
133
134 prog = sosineq(prog,lambda-epsi); % petersen's lemma's
    lambda constraint
135
136 % STEP 1 solution
137 options.solver = 'mosek';
138 sol = sossolve(prog,options);
139
140 Q0 = sosgetsol(sol,Q);
141 R0 = double(sosgetsol(sol,R));
142 S0 = sosgetsol(sol,S);
143 L = sosgetsol(sol, L);
144
145 T = sosgetsol(sol,T);
146 Delta_c = S0*inv(R0)*(S0')-Q0;
```

```
147
148 prog2 = sosprogram(vars);
149 prog2 = sosineq(prog2,Delta_c);
150 sol2 = sossolve(prog2,options);
151
152 if abs(sol2.solinfo.info.feasratio-1) <= 0.1   % Stop
    criteria: Delta_c test
153     K = -inv(R0)*(S0')            % Control gains
154     V = sosgetsol(sol,V)     % Lyapunov function
155 else
156     % STEP 2: Iterative method for V, Tx, alpha, Q, S, R,
        lambda
157     while k <= kmax
158         prog = sosprogram(vars);
159
160         [prog, Q] = sospolyvar(prog,monomials(x,2),'wscoeff
            ');
161         [prog, S] = sospolyvar(prog,monomials(x,1),'wscoeff
            ');
162         [prog, R] = sospolymatrixvar(prog,monomials(x,0),[m
            ,m],'symmetric');
163         [prog, V] = sospolyvar(prog,[monomials(x,4)],'
            wscoeff');
164         [prog, T] = sospolyvar(prog,[monomials(x,2)],'
            wscoeff');
165         [prog, lambda] = sospolyvar(prog,[monomials(x,0)],'
            wscoeff');
166         [prog, alpha] = sospolymatrixvar(prog,[monomials(x
            ,2)],[n+m+1,n+m+1],'symmetric');
167
168         % SOS constraints
169         prog = sosineq(prog,V-beta_V*(x1^2+x2^2)^n_V);
```

```
170         prog = sosineq(prog,T-beta_T*(x1^2+x2^2)^n_T);
171         gradV = [diff(V,x1); diff(V,x2)];   % gradient of
               Lyapunov function
172
173         Sigma1 = [gradV'*zeta_N'*Z+T-Q (1/2)*gradV'*zeta_M
               '*W-S
174                 ((1/2)*gradV'*zeta_M'*W-S)' -R];
175         Sigma2 = [(1/2)*Z'*Ad_i_N'; (1/2)*W'*Ad_i_M
               ']*[(1/2)*Ad_i_N*Z (1/2)*Ad_i_M*W];
176         B_aux = [gradV'*Qd^(1/2)
177               zeros(m,n)];
178
179         stability = [Sigma1+lambda*Sigma2 B_aux
                        % data-driven dissipativity-based
               condition for local stability
180                       B_aux' -lambda*eye(n)]+alpha*(1-L);
181
182         prog = sosineq(prog,-stability);
183
184         prog = sosineq(prog,R);                % dissipativity
               's R constraint
185
186         prog = sosineq(prog,R0-R);             % increasing
               delta constraints
187         delta_increasing = S*inv(R0)*(S0') + inv(R0)*(S0')
               '*S' - 2*S0*inv(R0)*(S0') + Q0 - Q;
188         prog = sosineq(prog,delta_increasing);
189
190         prog = sosineq(prog,lambda-epsi); % petersen's
               lemma's lambda constraint
191         prog = sosineq(prog,alpha);
192
```

```
193         % STEP 2 solution
194         sol = sossolve(prog,options);
195 
196         Q = sosgetsol(sol,Q)
197         R = double(sosgetsol(sol,R))
198         S = sosgetsol(sol,S)
199         alpha = sosgetsol(sol,alpha);
200         lambda = sosgetsol(sol,lambda);
201 
202         T = sosgetsol(sol,T);
203         Delta_c = S*inv(R)*(S')-Q;
204 
205         prog2 = sosprogram(vars);
206         prog2 = sosineq(prog2,Delta_c);
207         sol2 = sossolve(prog2,options);
208 
209         if abs(sol2.solinfo.info.feasratio-1) <= 0.1 | k ==
                kmax    % Stop criteria: Delta_c test
210             K = -inv(R)*(S')                              %
                   Control gains
211             V = sosgetsol(sol,V)                  % Lyapunov
                   function
212             break
213         end
214         k=k+1
215         Q0 = Q; R0 = R; S0 = S;
216     end
217 end
218 toc
219 
220 % Phase diagram - Open loop
221 syms x1 x2
```

```
222 a=10;
223 [x1,x2] = meshgrid(-a:0.1:a,-a:0.1:a);
224
225 dx1 = x1.^2 - x1.^3 + x2;
226 dx2 = 0.*x1;
227
228 figure
229 phase_OL = streamslice(x1,x2,dx1,dx2,1);
230 set(phase_OL,'LineWidth',1.5);
231 title('Open-loop phase diagram')
232 xlabel('$x_1$','Interpreter','latex'), ylabel('$x_2$','
       Interpreter','latex')
233 grid on
234 %%
235 % Phase diagram - Closed loop
236 syms x1 x2
237 global Kx
238 Kx = matlabFunction(p2s(K));
239 a=2;
240 passo = 0.01;
241 [x1,x2] = meshgrid(-a:passo:a,-a:passo:a);
242
243 dx1 = x1.^2 - x1.^3 + x2;
244 dx2 = Kx(x1,x2);
245
246 figure
247 phase_CL = streamslice(x1,x2,dx1,dx2,1.5);
248 set(phase_CL,'LineWidth',1.5);
249 title('Closed-loop phase diagram')
250 xlabel('$x_1$','Interpreter','latex'), ylabel('$x_2$','
       Interpreter','latex')
251 hold on
```

```
252 plot(0,0,'x','LineWidth',2)
253 grid on
254 elipL = matlabFunction(p2s(1-L));
255 zhandle = fimplicit(elipL);
256 zhandle.LineWidth = 1.5;
257 zhandle.LineStyle = "-";
258 zhandle.Color = "b";
259 x0 = [1; 1]; % initial condition
260 [t, xsol] = ode45(@experiment_CL, [0 15], x0, opt);
261 plot(xsol(:,1),xsol(:,2),'LineWidth',2);
262 figure
263 plot(t,xsol(:,1),t,xsol(:,2),'LineWidth',2);
264 title('Time response - closed-loop')
265 xlabel('t'); ylabel('x');
266 grid on
267 legend('$x_1$','$x_2$','Interpreter','latex','Location','
       northeast');
268
269 %------------------------------ EXPERIMENT
       --------------------------------%
270 function dx = experiment(t,x)
271     global delta
272
273     u = 5*sin(5*t);
274     dx = [x(1)^2-x(1)^3+x(2);
275           u];
276 end
277
278 function dx = experiment_CL(t,x)
279     global Kx
280     dx = [x(1)^2-x(1)^3+x(2);
281           Kx(x(1),x(2))];
```

```
282 end
```

Code 3 – Control design algorithm - system 2 - supply rate with constant matrices Q, S, R

```
1  % % % Data-driven Control - Dissipativity-based with
       Petersen's Lemma
2  % % % Author: Joao Gabriel Napoleao Silva - NEACON - UFC
3  % % % System: Van der Pol
4
5  clc; clear; close all
6
7  % Data-driven Dissipativity Polynomial Control
8  pvar x1 x2
9  vars = [x1;x2];
10 x = [x1;x2];
11
12 % e = 1;
13 % f = [x2;
14 %      -x1 + e*(1-x1^2)*x2];
15 %
16 % g = [0 1];
17 %
18 h = monomials(x,[1,3]);
19
20 m = 1; % u columns
21 n = 2; % x columns
22 p = size(h,1); % h length
23
24 %------------------------------ DATA COLLECT
       -------------------------------%
25 global delta
26 delta = 1e-2;    % Disturbance amplitude
```

```
27
28 x0 = [-0.2; 0.1]; % initial condition
29 opt = odeset('RelTol', 1e-3, 'AbsTol', 1e-6, 'NormControl',
       'off');
30 [t, xsol] = ode45(@vander, [0 10], x0, opt);
31 X00 = xsol';
32
33 U00 = sin(2*pi*t)'; % control signal
34 X11 = [X00(2,:)+sqrt(delta/n)*cos(2*pi*0.4*t)';
35       -X00(1,:)+X00(2,:)-X00(2,:).*(X00(1,:).^2)+U00+sqrt(
           delta/n)*sin(2*pi*0.4*t)'];
36
37 Ts = 100; % number of samples
38
39 X0 = X00(:,1:Ts);    % selecting first T samples of the
       experiment
40 X1 = X11(:,1:Ts);
41 U0 = U00(:,1:Ts);
42 t = t(1:Ts);
43
44 DELTA = sqrt(Ts*delta)*eye(n); %Disturbance upper bound
45
46 % Initialization of regressors
47 Z = [monomials(x,1:3)];
48 W = monomials(x,0);
49
50 N = length(Z);
51 M = length(W);
52
53 % plot of generated data
54 figure
55 subplot(2,1,1)
```

```
56 plot(t,U0,'LineWidth',1.5)
57 title('Input')
58 ylabel('u')
59 grid on
60
61 subplot(2,1,2)
62 plot(t,X0(1,:),'--',t,X0(2,:),'-.','LineWidth',1.5)
63 legend('$x_1$','$x_2$','Interpreter','latex','Location','
      northeast');
64 title('Open-loop response')
65 ylabel('x')
66 xlabel('t')
67 grid on
68
69 % construction of the data matrices Ad, Bd, Cd
70 Z0 = double(subs(Z,x,X0));
71 W0 = U0;
72
73 Ad = [Z0;W0]*[Z0;W0]';
74 Bd = -[Z0;W0]*X1';
75 Cd = X1*X1'-DELTA*DELTA';
76
77 % construction of the data matrices zeta and Q
78 zeta = -Ad\Bd;
79 Qd = Bd'*(Ad\Bd)-Cd;
80
81 zeta_N = zeta(1:N,:);
82 zeta_M = zeta((N+1):(N+M),:);
83 Ad_i = Ad^(-1/2);
84 Ad_i_N = Ad_i(:,1:N);
85 Ad_i_M = Ad_i(:,(N+1):(N+M));
86
```

```
87 % Inputs (design parameters)
88 beta_V = 1e-6;
89 beta_T = 1e-6;
90 beta_L = 1e-4;
91 n_V = 1;
92 n_T = 1;
93 n_L = 1;
94 epsi = 1e-8;
95 alpha = 1e-4*(x1^4+x2^4)*eye(n+m+1);
96
97 % Maximum iteration number
98 k = 1;
99 kmax = 50;
100
101 tic
102 % STEP 1: Determine V0, Tx, L0, Q0, S0, R0, lambda0
103 prog = sosprogram(vars);
104
105 [prog, Q] = sospolymatrixvar(prog,monomials(x,0),[p,p],'
    symmetric');
106 [prog, S] = sospolymatrixvar(prog,monomials(x,0),[p,m]);
107 [prog, R] = sospolymatrixvar(prog,monomials(x,0),[m,m],'
    symmetric');
108 [prog, V] = sospolyvar(prog,[monomials(x,2:4)],'wscoeff');
109 [prog, T] = sospolyvar(prog,[monomials(x,2)],'wscoeff');
110 [prog, lambda] = sospolyvar(prog,[monomials(x,0)],'wscoeff'
    );
111 [prog, L] = sospolyvar(prog,[monomials(x,2:4)],'wscoeff');
112
113 % SOS constraints
114 prog = sosineq(prog,V-beta_V*(x1^2+x2^2)^n_V); % V radially
     unbounded constraint
```

```
115 prog = sosineq(prog,T-beta_T*(x1^2+x2^2)^n_T); % T radially
        unbounded constraint
116 prog = sosineq(prog,L-beta_L*(x1^2+x2^2)^n_L);
117
118 gradV = jacobian(V,x)'; % gradient of Lyapunov function
119
120 Sigma1 = [gradV'*zeta_N'*Z+T-h'*Q'*h -h'*S+(1/2)*gradV'*
       zeta_M'*W;
121           (-h'*S+(1/2)*gradV'*zeta_M'*W)' -R];
122 Sigma2 = [(1/2)*Z'*Ad_i_N'; (1/2)*W'*Ad_i_M']*[(1/2)*Ad_i_N
       *Z (1/2)*Ad_i_M*W];
123 B_aux = [gradV'*Qd^(1/2);
124         zeros(m,n)];
125
126 stability = [Sigma1+lambda*Sigma2 B_aux           % data-
       driven dissipativity-based condition for local stability
127              B_aux' -lambda*eye(n)]+alpha*(1-L);
128
129 prog = sosineq(prog,-stability);
130
131 prog = sosineq(prog,R);      % dissipativity's R constraint
132
133 prog = sosineq(prog,lambda-epsi); % petersen's lemma's
       lambda constraint
134
135 % STEP 1 solution
136 options.solver = 'mosek';
137 sol = sossolve(prog,options);
138
139 Q0 = double(sosgetsol(sol,Q));
140 R0 = double(sosgetsol(sol,R));
141 S0 = double(sosgetsol(sol,S));
```

```
L = sosgetsol(sol, L);

T = sosgetsol(sol,T);
Delta_c = S0*(R0\(S0'))-Q0;
min(eig(Delta_c))

if min(eig(Delta_c)) >= 0     % Stop criterion: Delta_c
   test
    K = -R0\(S0')             % Control gains
    V = sosgetsol(sol,V)      % Lyapunov function
else
    % STEP 2: Iterative method for V, Tx, alpha, Q, S, R,
       lambda
    while k <= kmax
        prog = sosprogram(vars);

        [prog, Q] = sospolymatrixvar(prog,monomials(x,0),[p
           ,p],'symmetric');
        [prog, S] = sospolymatrixvar(prog,monomials(x,0),[p
           ,m]);
        [prog, R] = sospolymatrixvar(prog,monomials(x,0),[m
           ,m],'symmetric');
        [prog, V] = sospolyvar(prog,[monomials(x,2:4)],'
           wscoeff');
        [prog, T] = sospolyvar(prog,[monomials(x,2)],'
           wscoeff');
        [prog, lambda] = sospolyvar(prog,[monomials(x,0)],'
           wscoeff');
        [prog, alpha] = sospolymatrixvar(prog,[monomials(x
           ,2:6)],[n+m+1,n+m+1],'symmetric');

        % SOS constraints
```

```
165         prog = sosineq(prog,V-beta_V*(x1^2+x2^2)^n_V); % V
               radially unbounded constraint
166         prog = sosineq(prog,T-beta_T*(x1^2+x2^2)^n_T); % T
               radially unbounded costraint
167
168         gradV = jacobian(V,x)'; % gradient of Lyapunov
               function
169
170         Sigma1 = [gradV'*zeta_N'*Z+T-h'*Q'*h -h'*S+(1/2)*
               gradV'*zeta_M'*W
171                 (-h'*S+(1/2)*gradV'*zeta_M'*W)' -R];
172         Sigma2 = [(1/2)*Z'*Ad_i_N'; (1/2)*W'*Ad_i_M
               ']*[(1/2)*Ad_i_N*Z (1/2)*Ad_i_M*W];
173         B_aux = [gradV'*Qd^(1/2)
174               zeros(m,n)];
175
176         stability = [Sigma1+lambda*Sigma2 B_aux          %
               data-driven dissipativity-based condition for
               local stability
177                     B_aux' -lambda*eye(n)]+alpha*(1-L);
178         prog = sosineq(prog,-stability);
179
180         prog = sosineq(prog,R);              % dissipativity
               's R constraint
181
182         prog = sosineq(prog,R0-R);           % increasing
               delta constraints
183         delta_increasing = S*(R0\(S0')) + (R0\S0')'*S' - 2*
               S0*(R0\(S0')) + Q0 - Q;
184         prog = sosineq(prog,delta_increasing);
185
186         prog = sosineq(prog,lambda-epsi);    % petersen's
```

```
            lemma's lambda constraint
187         prog = sosineq(prog,alpha);
188
189         % STEP 2 solution
190         sol = sossolve(prog,options);
191
192         Q = double(sosgetsol(sol,Q))
193         R = double(sosgetsol(sol,R))
194         S = double(sosgetsol(sol,S))
195         alpha = sosgetsol(sol,alpha);
196         lambda = sosgetsol(sol,lambda);
197
198         T = sosgetsol(sol,T);
199         Delta_c = S*(R\(S'))-Q;
200         min(eig(Delta_c))
201         if min(eig(Delta_c))>= 0 | k == kmax      % Stop
               criterion: Delta_c test
202             K = -R\(S')                          % Control
                   gains
203             V = sosgetsol(sol,V)                 % Lyapunov
                   function
204             break
205         end
206         k=k+1
207         Q0 = Q; R0 = R; S0 = S;
208     end
209 end
210 toc
211
212 % Phase diagram - Open loop
213 syms x1 x2
214 a=5;
```

```
215 [x1,x2] = meshgrid(-a:0.1:a,-a:0.1:a);
216 e=1;
217 dx1 = x2;
218 dx2 = -x1 + e.*(1-x1.^2).*x2;
219
220 figure
221 OL = streamslice(x1,x2,dx1,dx2,1);
222 set(OL,'LineWidth',1.5);
223 title('Phase diagram - open-loop')
224 xlabel('$x_1$','Interpreter','latex'), ylabel('$x_2$','
       Interpreter','latex')
225 grid on
226
227 %%
228 % % Phase diagram - Closed loop
229 uk = K*h;
230 global uf
231 uf = matlabFunction(p2s(uk));
232 a=1;
233 [x1,x2] = meshgrid(-a:0.1:a,-a:0.1:a);
234 dx1 = x2;
235 dx2 = -x1 + e.*(1-x1.^2).*x2 + uf(x1,x2);
236
237 figure
238 CL = streamslice(x1,x2,dx1,dx2,4);
239 set(CL,'LineWidth',1.5);
240 title('Phase diagram - closed-loop')
241 xlabel('$x_1$','Interpreter','latex'), ylabel('$x_2$','
       Interpreter','latex')
242 hold on
243 plot(0,0,'x','LineWidth',2)
244 grid on
```

```
245
246 ellipL = fimplicit(p2s(1-L));
247 ellipL.LineWidth = 1;
248 ellipL.Color = 'b';
249 x0 = [-0.3; -0.4]; % initial condition
250 [t, xsol] = ode45(@vander_cl, [0 10], x0, opt);
251 plot(xsol(:,1),xsol(:,2),'LineWidth',2);
252 figure
253 plot(t,xsol(:,1),t,xsol(:,2),'LineWidth',2);
254 title('Time response - closed-loop')
255 xlabel('t'); ylabel('x');
256 grid on
257 legend('$x_1$','$x_2$','Interpreter','latex','Location','
      northeast');
258
259 %------------------------------ EXPERIMENT
      --------------------------------%
260 function dx = vander(t,x)
261     e=1;
262     global delta
263     u = sin(2*pi*t);
264     dx = [x(2);
265           -x(1) + e*(1-x(1)^2)*x(2) + u];
266 end
267
268 function dx = vander_cl(t,x)
269     global uf
270     e=1;
271     dx = [x(2);
272           -x(1) + e*(1-x(1)^2)*x(2) + uf(x(1),x(2))];
273 end
```

Code 4 – Control design algorithm - system 2 - supply rate with polynomial matrices Q, S, R

```
% % % Data-driven Control - Dissipativity-based with Petersen's Lemma
% % % Author: Joao Gabriel Napoleao Silva - NEACON - UFC
% % % System: Van der Pol

clc; clear; close all

% Data-driven Dissipativity Polynomial Control
pvar x1 x2
vars = [x1;x2];
x = [x1;x2];

% e = 1;
% f = [x2;
%      -x1 + e*(1-x1^2)*x2];
%
% g = [0 1];


m = 1; % u columns
n = 2; % x columns

%------------------------------ DATA COLLECT ------------------------------%
global delta
delta = 1e-2;   % Disturbance amplitude

x0 = [-0.2; 0.1]; % initial condition
opt = odeset('RelTol', 1e-3, 'AbsTol', 1e-6, 'NormControl', 'off');
[t, xsol] = ode45(@vander, [0 10], x0, opt);
```

```
X00 = xsol';
U00 = sin(2*pi*t)'; % control signal
X11 = [X00(2,:)+sqrt(delta/n)*cos(2*pi*0.4*t)';
      -X00(1,:)+X00(2,:)-X00(2,:).*(X00(1,:).^2)+U00+sqrt(
         delta/n)*sin(2*pi*0.4*t)'];

Ts = 100; % number of samples

X0 = X00(:,1:Ts);    % selecting first T samples of the
   experiment
X1 = X11(:,1:Ts);
U0 = U00(:,1:Ts);
t = t(1:Ts);

DELTA = sqrt(Ts*delta)*eye(n); %Disturbance upper bound

% Initialization of regressors
Z = [monomials(x,1:3)];
W = monomials(x,0);

N = length(Z);
M = length(W);

% plot of generated data
figure
subplot(2,1,1)
plot(t,U0,'LineWidth',1.5)
title('Input')
ylabel('u')
grid on

subplot(2,1,2)
```

```
59 plot(t,X0(1,:),'--',t,X0(2,:),'-.','LineWidth',1.5)
60 legend('$x_1$','$x_2$','Interpreter','latex','Location','
      northeast');
61 title('Open-loop response')
62 ylabel('x')
63 xlabel('t')
64 grid on
65
66 % construction of the data matrices Ad, Bd, Cd
67 Z0 = double(subs(Z,x,X0));
68 W0 = U0;
69
70 Ad = [Z0;W0]*[Z0;W0]';
71 Bd = -[Z0;W0]*X1';
72 Cd = X1*X1'-DELTA*DELTA';
73
74 % construction of the data matrices zeta and Q
75 zeta = -Ad\Bd;
76 Qd = Bd'*(Ad\Bd)-Cd;
77
78 zeta_N = zeta(1:N,:);
79 zeta_M = zeta((N+1):(N+M),:);
80 Ad_i = Ad^(-1/2);
81 Ad_i_N = Ad_i(:,1:N);
82 Ad_i_M = Ad_i(:,(N+1):(N+M));
83
84 % Inputs (design parameters)
85 beta_V = 1e-6;
86 beta_T = 1e-6;
87 beta_L = 1e-4;
88 n_V = 1;
89 n_T = 1;
```

```
n_L = 1;
epsi = 1e-8;
alpha = 1e-4*(x1^4+x2^4)*eye(n+m+1);

% Maximum iteration number
k = 1;
kmax = 50;

tic
% STEP 1: Determine V0, Tx, L0, Q0, S0, R0, lambda0
prog = sosprogram(vars);

[prog, Q] = sospolyvar(prog,monomials(x,[2:6]),'wscoeff');
[prog, S] = sospolyvar(prog,monomials(x,[1,3]),'wscoeff');
[prog, R] = sospolymatrixvar(prog,monomials(x,0),[m,m],'
   symmetric');
[prog, V] = sospolyvar(prog,[monomials(x,2:4)],'wscoeff');
[prog, T] = sospolyvar(prog,[monomials(x,2)],'wscoeff');
[prog, lambda] = sospolyvar(prog,[monomials(x,0)],'wscoeff'
   );
[prog, L] = sospolyvar(prog,[monomials(x,2:4)],'wscoeff');

% SOS constraints
prog = sosineq(prog,V-beta_V*(x1^2+x2^2)^n_V); % V radially
    unbounded constraint
prog = sosineq(prog,T-beta_T*(x1^2+x2^2)^n_T); % T radially
    unbounded constraint
prog = sosineq(prog,L-beta_L*(x1^2+x2^2)^n_L);

gradV = jacobian(V,x)'; % gradient of Lyapunov function

Sigma1 = [gradV'*zeta_N'*Z+T-Q -S+(1/2)*gradV'*zeta_M'*W;
```

```
118           (-S+(1/2)*gradV'*zeta_M'*W)' -R];
119 Sigma2 = [(1/2)*Z'*Ad_i_N'; (1/2)*W'*Ad_i_M']*[(1/2)*Ad_i_N
    *Z (1/2)*Ad_i_M*W];
120 B_aux = [gradV'*Qd^(1/2);
121        zeros(m,n)];
122
123 stability = [Sigma1+lambda*Sigma2 B_aux          % data-
    driven dissipativity-based condition for local stability
124              B_aux' -lambda*eye(n)]-alpha*(1-L);
125
126 prog = sosmatrixineq(prog,-stability);
127
128 prog = sosineq(prog,R);     % dissipativity's R constraint
129
130 prog = sosineq(prog,lambda-epsi); % petersen's lemma's
    lambda constraint
131
132 % STEP 1 solution
133 options.solver = 'mosek';
134 sol = sossolve(prog,options);
135
136 Q0 = sosgetsol(sol,Q);
137 R0 = double(sosgetsol(sol,R));
138 S0 = sosgetsol(sol,S);
139 L = sosgetsol(sol, L);
140
141 T = sosgetsol(sol,T);
142 Delta_c = S0*inv(R0)*(S0')-Q0;
143
144 prog2 = sosprogram(vars);
145 prog2 = sosineq(prog2,Delta_c);
146 sol2 = sossolve(prog2,options);
```

```
147
148 if abs(sol2.solinfo.info.feasratio-1) <= 0.1     % Stop
    criterion: Delta_c test
149     K = -inv(R0)*S0'            % Control gains
150     V = sosgetsol(sol,V)    % Lyapunov function
151 else
152     % STEP 2: Iterative method for V, Tx, alpha, Q, S, R,
        lambda
153     while k <= kmax
154         prog = sosprogram(vars);
155
156         [prog, Q] = sospolyvar(prog,monomials(x,[2:6]),'
            wscoeff');
157         [prog, S] = sospolyvar(prog,monomials(x,[1,3]),'
            wscoeff');
158         [prog, R] = sospolymatrixvar(prog,monomials(x,0),[m
            ,m],'symmetric');
159         [prog, V] = sospolyvar(prog,[monomials(x,2:4)],'
            wscoeff');
160         [prog, T] = sospolyvar(prog,[monomials(x,2)],'
            wscoeff');
161         [prog, lambda] = sospolyvar(prog,[monomials(x,0)],'
            wscoeff');
162         [prog, alpha] = sospolymatrixvar(prog,[monomials(x
            ,2:6)],[n+m+1,n+m+1],'symmetric');
163
164         % SOS constraints
165         prog = sosineq(prog,V-beta_V*(x1^2+x2^2)^n_V); % V
            radially unbounded constraint
166         prog = sosineq(prog,T-beta_T*(x1^2+x2^2)^n_T); % T
            radially unbounded costraint
167
```

```
168         gradV = jacobian(V,x)'; % gradient of Lyapunov
               function
169
170         Sigma1 = [gradV'*zeta_N'*Z+T-Q -S+(1/2)*gradV'*
               zeta_M'*W
171                 (-S+(1/2)*gradV'*zeta_M'*W)' -R];
172         Sigma2 = [(1/2)*Z'*Ad_i_N'; (1/2)*W'*Ad_i_M
               ']*[(1/2)*Ad_i_N*Z (1/2)*Ad_i_M*W];
173         B_aux = [gradV'*Qd^(1/2)
174               zeros(m,n)];
175
176         stability = [Sigma1+lambda*Sigma2 B_aux          %
               data-driven dissipativity-based condition for
               local stability
177                     B_aux' -lambda*eye(n)]-alpha*(1-L);
178         prog = sosineq(prog,-stability);
179
180         prog = sosineq(prog,R);              % dissipativity
               's R constraint
181
182         prog = sosineq(prog,R0-R);           % increasing
               delta constraints
183         delta_increasing = R*inv(R0)*(S0') + inv(R0)*(S0')
               '*S' - 2*S0*inv(R0)*(S0') + Q0 - Q;
184         prog = sosineq(prog,delta_increasing);
185
186         prog = sosineq(prog,lambda-epsi);    % petersen's
               lemma's lambda constraint
187         prog = sosineq(prog,alpha);
188
189         % STEP 2 solution
190         sol = sossolve(prog,options);
```

```
191
192         Q = sosgetsol(sol,Q)
193         R = double(sosgetsol(sol,R))
194         S = sosgetsol(sol,S)
195         alpha = sosgetsol(sol,alpha);
196         lambda = sosgetsol(sol,lambda);
197
198         T = sosgetsol(sol,T);
199         Delta_c = S*inv(R)*(S')-Q;
200
201         prog2 = sosprogram(vars);
202         prog2 = sosineq(prog2,Delta_c);
203         sol2 = sossolve(prog2,options);
204         if abs(sol2.solinfo.info.feasratio-1) <= 0.1 | k ==
                kmax     % Stop criterion: Delta_c test
205             K = -inv(R)*S'                                  %
                   Control gains
206             V = sosgetsol(sol,V)                  % Lyapunov
                   function
207             break
208         end
209         k=k+1
210         Q0 = Q; R0 = R; S0 = S;
211     end
212 end
213 toc
214
215 % Phase diagram - Open loop
216 syms x1 x2
217 a=5;
218 [x1,x2] = meshgrid(-a:0.1:a,-a:0.1:a);
219 e=1;
```

```
dx1 = x2;
dx2 = -x1 + e.*(1-x1.^2).*x2;

figure
OL = streamslice(x1,x2,dx1,dx2,1);
set(OL,'LineWidth',1.5);
title('Phase diagram - open-loop')
xlabel('$x_1$','Interpreter','latex'), ylabel('$x_2$','
   Interpreter','latex')
grid on

%%
% % Phase diagram - Closed loop
uk = K;
global uf
uf = matlabFunction(p2s(uk));
a=5;
[x1,x2] = meshgrid(-a:0.1:a,-a:0.1:a);
dx1 = x2;
dx2 = -x1 + e.*(1-x1.^2).*x2 + uf(x1,x2);

figure
CL = streamslice(x1,x2,dx1,dx2,4);
set(CL,'LineWidth',1.5);
title('Phase diagram - closed-loop')
xlabel('$x_1$','Interpreter','latex'), ylabel('$x_2$','
   Interpreter','latex')
hold on
plot(0,0,'x','LineWidth',2)
grid on

ellipL = fimplicit(p2s(1-L));
```

```
ellipL.LineWidth = 1;
ellipL.Color = 'b';
x0 = [2; 2]; % initial condition
[t, xsol] = ode45(@vander_cl, [0 300], x0, opt);
plot(xsol(:,1),xsol(:,2),'LineWidth',2);
figure
plot(t,xsol(:,1),t,xsol(:,2),'LineWidth',2);
title('Time response - closed-loop')
xlabel('t'); ylabel('x');
grid on
legend('$x_1$','$x_2$','Interpreter','latex','Location','
   northeast');

%------------------------------ EXPERIMENT
   --------------------------------%
function dx = vander(t,x)
    e=1;
    global delta
    u = sin(2*pi*t);
    dx = [x(2) + sqrt(delta)*cos(2*pi*0.4*t);
         -x(1) + e*(1-x(1)^2)*x(2) + u + sqrt(delta)*sin(2*
            pi*0.4*t)];
end

function dx = vander_cl(t,x)
    global uf
    e=1;
    dx = [x(2);
         -x(1) + e*(1-x(1)^2)*x(2) + uf(x(1),x(2))];
end
```