May 2019

# Advances in Reachability Analysis for Nonlinear Dynamic Systems

Kai Shen
*Clemson University*, shenkaiusa@gmail.com

# ADVANCES IN REACHABILITY ANALYSIS FOR NONLINEAR DYNAMIC SYSTEMS

---

A Dissertation
Presented to
the Graduate School of
Clemson University

---

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
Chemical Engineering

---

by
Kai Shen
May 2019

---

Accepted by:
Dr. Joseph K. Scott, Committee Chair
Dr. Marc R. Birtwistle
Dr. Eric M. Davis
Dr. Yue Wang

# Abstract

Systems of nonlinear ordinary differential equations (ODEs) are used to model an incredible variety of dynamic phenomena in chemical, oil and gas, and pharmaceutical industries. In reality, such models are nearly always subject to significant uncertainties in their initial conditions, parameters, and inputs.

This dissertation provides new theoretical and numerical techniques for rigorously enclosing the set of solutions reachable by a given systems of nonlinear ODEs subject to uncertain initial conditions, parameters, and time-varying inputs. Such sets are often referred to as reachable sets, and methods for enclosing them are critical for designing systems that are passively robust to uncertainty, as well as for optimal real-time decision-making. Such enclosure methods are used extensively for uncertainty propagation, robust control, system verification, and optimization of dynamic systems arising in a wide variety of applications.

Unfortunately, existing methods for computing such enclosures often provide an unworkable compromise between cost and accuracy. For example, interval methods based on differential inequalities (DI) can produce bounds very efficiently but are often too conservative to be of any practical use. In contrast, methods based on more complex sets can achieve sharp bounds, but are far too expensive for real-time decision-making and scale poorly with problem size.

Recently, it has been shown that bounds computed via differential inequalities

can often be made much less conservative while maintaining high efficiency by exploiting redundant model equations that are known to hold for all trajectories of interest (e.g., linear relationships among chemical species in a reaction network that hold due to the conservation of mass or elements). These linear relationships are implied by the governing ODEs, and can thus be considered redundant. However, these advances are only applicable to a limited class of system in which pre-existing linear redundant model equations are available. Moreover, the theoretical results underlying these algorithms do not apply to redundant equations that depend on time-varying inputs and rely on assumptions that prove to be very restrictive for nonlinear redundant equations, etc.

This dissertation continues a line of research that has recently achieved very promising bounding results using methods based on differential inequalities. In brief, the major contributions can be divided into three categories: (1) In regard to algorithms, this dissertation significantly improves existing algorithms that exploit linear redundant model equations to achieve more accurate and efficient enclosures. It also develops new fast and accurate bounding algorithms that can exploit nonlinear redundant model equations. (2) Considering theoretical contributions, it develops a novel theoretical framework for the introduction of redundant model equations into arbitrary dynamic models to effectively reduce conservatism. The newly developed theories have more generality in terms of application. For example, complex nonlinear constraints that involve states, time derivatives of the system states, and time-varying inputs are allowed to be exploited. (3) A new differential inequalities method called Mean Value Differential Inequalities (MVDI) is developed that can automatically introduce redundant model equations for arbitrary dynamic systems and has a second-order convergence rate reported the first time among DI-based methods.

# Dedication

First of all, I would like to express my gratitude to my advisor Professor Joseph Kirk Scott. This thesis could not have been finished without his patient and intelligent support. The depth and breadth of his knowledge always inspires me. He brought me into a new world and changed my old knowledge structure. I am deeply grateful for the freedom that he gave me to pursue exciting topics. He is a mentor, a friend, and an excellent educator.

I want to thank my thesis committee, Professors Marc Birtwistle, Eric Davis, and Yue Wang for their help and support. I would also like to acknowledge the support of my laboratory members. My improvement has been made by many tangential conversations with them. Special thanks to Yuanxun Shao, Alphonse Hakizimana, Dillard Robertson, Taehun Kim, and Xuejiao Yang.

I always feel lucky to have a hospitable and loving wife. Wherever we are, that is our home, and we always enjoy spending time together. Thanks to her for taking care of our two lovely little daughters so that I can focus on my day-to-day research. I am grateful that she gave up many things to support my career.

Finally, to my parents, what can I say? Your dedication to my education and selfless love led me to this point. I am proud of being your son.

# Table of Contents

# List of Figures

# Chapter 1

# Introduction

A huge variety of dynamic phenomena in chemical, oil and gas, and pharmaceutical industries can be modeled by systems of nonlinear ordinary differential equations (ODEs). In applications from the biochemical networks inside a cell to unit operations in a chemical plant, these mathematical models are nearly always subject to significant uncertainties in their initial conditions, model parameters, and inputs.

The ability to quantify the effects of these uncertainties on the model solution is essential for making optimal real-time decisions in complex, uncertain environments, as well as for designing systems that are passively robust to uncertainty. Quantifying uncertainty in terms of rigorous enclosures of the system states achievable under uncertainty is uniquely useful for safety verification processes, which can guarantee that a system will always satisfy all constraints (e.g., final product specifications, overheat protection in a chemical reactor, etc.).

Although it has long been possible to compute such enclosures, existing methods often provide an unworkable trade-off between computational cost and enclosure tightness. Interval methods based on differential inequalities can compute enclosures

very efficiently, with a computational cost comparable to integrating a small number of single trajectories, but the enclosures are often too conservative to be used in practice. In contrast, modern bounding methods that use more complex sets such as zonotopes or ellipsoids can achieve tight bounds even for systems with high nonlinearity and large uncertainties, but are too expensive for real-time decision-making for most practical systems. Therefore, it is critical to develop an alternative approach that can generate accurate rigorous enclosures that are fast enough for real-time applications and scalable to large-scale systems.

The general objective of this dissertation is to develop theoretical and numerical methods for uncertainty quantification in dynamic systems. In particular, novel methods are developed for rapidly computing rigorous and accurate enclosures of the solutions of nonlinear ordinary differential equations subject to bounded initial conditions, parameters, and time-varying inputs.

In the remainder of this chapter, reachable set enclosures are discussed and motivated in more detail, and then the core contributions of this dissertation are summarized.

## 1.1 Enclosures of Reachable Sets

Consider a dynamic system described by the following ordinary differential equations (ODEs):

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{u}(t), \mathbf{x}(t)), \tag{1.1a}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0. \tag{1.1b}$$

Let $I = [t_0, t_f] \subset \mathbb{R}$ be a time horizon of interest, and let $X_0$ and $\mathcal{U}$ be sets of admissible initial conditions and inputs, respectively.

**Definition 1** *The* reachable set *of the system* (1.1) *is defined for every* $t \in I$ *as*

$$Re(t) \equiv \{\mathbf{x}(t) \in \mathbb{R}^{n_x} : \mathbf{x} \textit{ is a solution of } (1.1) \textit{ for some } (\mathbf{x}_0, \mathbf{u}) \in X_0 \times \mathcal{U}\}. \quad (1.2)$$

As the above definition shows, the reachable set is the set of all states reachable by a given system of nonlinear ordinary differential equations (ODEs) subject to uncertain initial conditions and model inputs. Computing the exact reachable set is very difficult for most systems in practice. Instead, a time-varying rigorous enclosure of $Re(t)$ is often computed.

Reachable set enclosures are useful for quantifying the effects of uncertainty in dynamic models arising in a variety of applications, including (bio)chemical reaction networks [58, 42], autonomous vehicles [80, 3], and power systems [51, 2]. Such methods are also widely used for control applications, where the reachable sets of interest describe the uncertainty in a systems future evolution arising from external disturbances, imprecisely known model parameters, and measurement errors. With the ability to capture the behavior of all possible trajectories, enclosing these sets is the central step in robust (i.e., set-based) state estimation [42, 54], which is in turn essential for robust model predictive control [32], fault detection [60, 35, 53], and safety verification [3, 14, 36]. Guaranteed conclusions are necessary in these applications because it is possible that even a million simulations will still miss some isolated but critical scenarios. With branch-and-bound approaches, such enclosures can also be used to compute robust design spaces for pharmaceutical process [29, 20]. Finally, reachable set enclosures are also useful for dynamic optimization, which has applications in parameter estimation [73], open-loop optimal control [46], aircraft/spacecraft

maneuvers [52], and batch chemical processes [75], to name only a few. In this context, the reachable sets of interest describe the range of solutions that can be achieved by decision variables lying in a given region of the search space, and enclosures are used to eliminate regions by proving infeasibility or suboptimality with certainty. When applied within a branch-and-bound framework, this enables the solution of dynamic optimization problems to guaranteed global optimality [62, 22, 33, 49].

Note that (1.1) and its reachable set are defined here for illustration and motivation. Later, each chapter has its own problem statement that is slightly different from (1.1) for technical reasons.

### 1.1.1 Existing Enclosing Methods

Although it has long been possible to compute rigorous enclosures, existing methods often cannot provide enclosures with sufficient speed and accuracy for many critical applications. For example, in set-based state estimation and robust control, the desired enclosures depend on process measurements. Thus, these applications require methods that are both fast enough for real-time implementation and accurate enough to be useful for decision-making. Similarly, global dynamic optimization requires accurate enclosures to avoid excessive subdivision of the search space, and high speed because even accurate methods may still need to consider thousands of regions [81].

Existing approaches for rigorously enclosing the reachable sets of nonlinear ODEs can be grouped into four broad approaches: level-set approaches, Taylor series methods, conservative linearization methods, and differential inequalities.

Level set approaches [41, 30] compute approximations of reachable sets by constructing and solving the Hamilton-Jacobi (HJ) partial differential equations (PDEs)

on grids which represent a discretization of state space. Although such methods can provide very accurate approximations of the reachable set, solving HJ PDE is intractable when the number of states exceeds $\sim$5.

Taylor series approaches propagate enclosures of the reachable set over discrete time steps by constructing a Taylor expansion of the state with respect to time and bounding the coefficients with, e.g., interval arithmetic (IA) [47]. The resulting enclosure is then inflated by a rigorous bound on the truncation error. Classical methods propagate interval enclosures, which makes them relatively efficient but often very conservative. In contrast, modern methods have achieved high accuracy in many applications by replacing interval bounds with Taylor models, which are multivariate Taylor expansions in the model inputs with rigorous interval remainder bounds [7, 34]. Further improvements have recently been achieved through the use of Taylor models with remainder bounds described by ellipsoids or more general sets [24, 23]. However, achieving high accuracy with these methods often requires high-order Taylor models, which can become intractable because the number of coefficients scales exponentially in the number of states and model inputs [7].

Conservative linearization approaches propagate enclosures of the reachable set over discrete time steps by first considering a locally linearized model and subsequently adding a rigorous bound on the linearization error [4, 5]. The key advantage of this approach is that the reachable set of the linearized system can be enclosed very accurately using efficient set representations such as ellipsoids or zonotopes. Modern methods of this type have been shown to produce highly accurate enclosures in many applications [2, 3]. However, like Taylor series approaches, this often requires very complex set representations and hence high computational cost (see, e.g., the use of 400th order zonotopes, each described by 2406 real numbers, to enclose a 6-dimensional reachable sets in [64]).

Finally, approaches based on differential inequalities (DI) operate by constructing an auxiliary system of ODEs, twice the size of the original, that describes componentwise upper and lower bounds on the reachable set as its solutions. Harrison [16] originally observed that such a system can be constructed automatically using simple interval arithmetic. Moreover, this system can be solved with state-of-the-art numerical integration codes, whereas both Taylor series and conservative linearization methods require custom integration algorithms with significant step-size restrictions [34, 5]. Thus, DI methods are capable of producing bounds very rapidly (i.e., at a small multiple of the cost of integrating a single trajectory [64, 59, 61]), making DI a potentially powerful tool for real-time control and global dynamic optimization. However, the resulting enclosures are often extremely conservative unless the ODEs satisfy restrictive monotonicity conditions [16]. Several methods have been proposed to address this by enabling the use of more complex reachable set representations in place of intervals. The work [9] proposes an interesting use of DI to compute Taylor model enclosures. However, auxiliary ODEs are required for each Taylor coefficient, which is prohibitive for high-order expansions. The article [77] introduces a general framework for using DI to compute general convex enclosures. Specific implementations can be found in [65, 63, 18]. In particular, the article [18] introduces a very effective method for computing polytopic enclosures using DI. However, this method creates an auxiliary system of ODEs whose right-hand sides are evaluated by solving embedded linear programs rather than using simple IA, which leads to significantly longer computation times.

To overcome these issues, several DI methods with greatly improved accuracy have also been developed, specifically for systems whose states are known to satisfy a set of state constraints pointwise in time [74, 58, 64, 17, 19]. Examples of such constraints include physically motivated upper and lower bounds, such as nonnegativity

of certain states [74], or algebraic functions of the states that are known to remain constant with time due to, e.g., the conservation of mass, energy, or chemical elements (we call such functions *solution invariants*) [64]. Constraints of this type are implied by the dynamics, and are therefore satisfied by all system trajectories. Invariants are redundant with the ODEs. In contrast, the article [19] considers state constraints that are externally imposed, such as path constraints in the context of dynamic optimization, where one is only interested in bounding the feasible trajectories. In both cases, enhanced DI methods have been developed that can exploit these constraints during the bounding procedure, often resulting in much tighter bounds with only a moderate increase in computational cost [64, 17, 19]. Many numerical examples have been shown that existing DI methods using pre-existing linear constraints are far superior to many modern bounding methods in regards to both accuracy and efficiency. In brief, this is accomplished by applying a suitably defined bound refinement operator pointwise in time during the forward propagation of the bounds. At each point in time, this refinement operator attempts to shrink the current bounds by eliminating enclosed regions that violate the constraints. However, an evident drawback of these DI methods is that they only apply to systems for which appropriate linear constraints are known *a priori*.

## 1.2 Contributions

In this dissertation, Chapters 2–4 are devoted to novel methods for computing accurate and efficient enclosures of the reachable sets of ODEs. Inspired by existing DI methods [64] that use pre-existing linear solution invariants to reduce the conservatism of the computed bounds, Chapter 2 presents a new framework for introducing 'manufactured invariants' into arbitrary dynamic systems to effectively reduce con-

servatism. The new framework deliberately augments general nonlinear systems with redundant states and differential equations such that the new augmented systems automatically have redundant model equations that can be exploited in the bounding procedure. Thus, this new framework has the potential to extend very effective DI methods for systems with invariants to general dynamic systems. Several guidelines are provided in Chapter 2 to help derive effective 'manufactured invariants' which require some problem insights. Besides this new framework, Chapter 2 also develops new strategies for further increasing the efficiency and reducing the conservatism of the bounding methods. In particular, new preconditioning techniques are developed to reformulate existing linear invariants so that the preconditioned invariants are superior to the original ones in terms of bound tightness. A faster algorithm that exploits linear invariants is also presented to reduce the computational cost of the refinement procedure. Many numerical examples across different applications clearly demonstrate that extremely effective manufactured invariants very often exist and they can be exploited to reduce conservatism, often dramatically, at modest additional cost.

In Chapter 3, new and significantly more powerful theorems that deal with nonlinear constraints, and corresponding refinement algorithms are developed. These are very important because, besides linear invariants, many practical systems actually have pre-existing nonlinear invariants (e.g., oscillators and Hamiltonian systems [71]). Moreover, nonlinear path constraints arise in a wide variety of optimal control problems [13]. In addition, manufactured nonlinear invariants can also be generated for arbitrary dynamic systems by using the technique introduced in Chapter 2. Moreover, the key DI theorem underlying existing methods does not permit the redundant invariants and/or constraints to depend on uncertain parameters or time-varying inputs in the model. Unfortunately, there are no existing DI theorems and algorithms

that can deal with these issues. Chapter 3 extends the existing DI methods by: (i) Addressing problems with nonlinear invariants/constraints that depend on uncertain time-varying inputs and state derivatives, which requires fundamentally new supporting theories; (ii) Generalizing the refinement algorithm to treat equality and inequality nonlinear constraints, rather than just linear constraints. The theoretical and algorithmic contributions outlined above significantly increase the applicability of state-of-the-art DI methods for computing sharp bounds on the solutions of uncertain nonlinear systems. Problems that have been investigated in other state-of-the-art bounding methods [77, 17, 35] are compared, and bounds with similar or even better accuracy but at a significantly reduced cost are achieved.

Until this point, our conclusion is that DI methods that exploit constraints can achieve sharp bounds at low cost compared with other modern methods. However, these achievements are made by using either pre-existing or manually derived redundant model equations. Although several mechanisms are outlined in Chapter 2 to help create effective 'manufactured invariants', some problem-specific insights are needed. To extend these methods to general nonlinear systems, techniques for automatically manufacturing effective nonlinear redundant model equations are still missing. Chapter 4 provides one way to automate the construction of redundant model equations. In particular, a new differential inequalities method called Mean Value Differential Inequalities (MVDI) is introduced. MVDI creates approximate algebraic relations between the original states and their parametric sensitivities by a first-order Taylor expansion. Such algebraic relations can be used in the refinement procedure introduced in Chapters 2 and 3 to achieve sharp bounds. However, since these algebraic relations only hold approximately, the methods developed in Chapter 3 cannot be used. This is because MVDI creates approximate rather than exact algebraic relations between the original states and sensitivities, and exploiting them

requires fundamentally different theory and algorithms than that just using invariants. The new theory in Chapter 4 is very useful in algorithms for globally solving optimal control problems. Besides that, a new DI algorithm is presented that combines refinements based on mean-value enclosures and existing state constraints, and also includes an improved method for bounding the right-hand sides of the given ODEs. Moreover, a new theory is developed for analyzing the accuracy of the computed bounds. Specifically, it proves that the new MVDI method satisfies a second-order convergence property with respect to the size of the uncertainty set, which has previously only been achieved for methods that use more complex sets such as Taylor models.

# Chapter 2

# Rapid and Accurate Reachability Analysis for Nonlinear Dynamic Systems by Exploiting Model Redundancy

## 2.1 Introduction

This chapter presents as future evolution arising from external disturbances, imprecisely known model parameters, and measuremen errors. Enclosing these sets is the central step in robust (i.e., set-based) state estimation [42, 54], which is in turn essential for robust model predictive control [32], guaranteed fault detection [60, 35, 53], and safety verification [3, 14]. Finally, reachable set enclosures are also useful for dynamic optimization, which has applications in parameter estimation [73], open-loop optimal control [46], aircraft/s new technique for rapidly and accurately computing a rigorous enclosure of the set of solutions reachable by a given system

of nonlinear ordinary differential equations (ODEs) subject to a given range of inputs (i.e., initial conditions and model parameters). Such sets are commonly referred to as reachable sets, and methods for enclosing them are useful for quantifying the effects of uncertainty in dynamic models arising in a variety of applications, including (bio)chemical reaction networks [58, 42], autonomous vehicles [80, 3], and power systems [51, 2]. Such methods are also widely used for control applications, where the reachable sets of interest describe the uncertainty in a systempacecraft maneuvers [52], and batch chemical processes [75], to name only a few. In this context, the reachable sets of interest describe the range of solutions that can be achieved by decision variables lying in a given region of the search space, and enclosures are used to eliminate regions by proving infeasibility or suboptimality with certainty. When applied within a branch-and-bound framework, this enables the solution of dynamic optimization problems to guaranteed global optimality [62, 22, 33, 49].

Unfortunately, exiting methods often cannot provide enclosures with sufficient speed and accuracy for many critical applications. Clearly, control applications require enclosure methods that are simultaneously fast enough to be used in real-time and accurate enough to be useful for decision-making [3, 55]. Similarly, global dynamic optimization codes require highly accurate enclosures to avoid excessive subdivision of the search space, but also require high speed because even the most accurate methods may still need to consider thousands of regions to solve practical problem instances [81]. For nonlinear systems of practical complexity, this combination of speed and accuracy remains a significant challenge.

Existing approaches for rigorously enclosing the reachable sets of nonlinear ODEs can be grouped into three broad approaches: Taylor series approaches, conservative linearization, and differential inequalities. Taylor series approaches propagate enclosures of the reachable set over discrete time steps by constructing a Taylor expan-

sion of the state with respect to time and bounding the coefficients with, e.g., interval arithmetic (IA) [47]. The resulting enclosure is then inflated by a rigorous bound on the truncation error. Classical methods propagate interval enclosures, which makes them relatively efficient but often very conservative. In contrast, modern methods have achieved high accuracy in many applications by replacing interval bounds with Taylor models, which are multivariate Taylor expansions in the model inputs with rigorous interval remainder bounds [7, 34]. Further improvements have recently been achieved through the use of Taylor models with remainder bounds described by ellipsoids or more general sets [24, 23]. However, achieving high accuracy with these methods often requires high-order Taylor models, which can become intractable because the number of coefficients scales exponentially in the number of states and model inputs [7].

Conservative linearization approaches propagate enclosures of the reachable set over discrete time steps by first considering a locally linearized model and subsequently adding a rigorous bound on the linearization error [4, 5]. The key advantage of this approach is that the reachable set of the linearized system can be enclosed very accurately using efficient set representations such as ellipsoids or zonotopes. Modern methods of this type have been shown to produce highly accurate enclosures in many applications [2, 3]. However, like Taylor series approaches, this often requires very complex set representations and hence high computational cost (see, e.g., the use of 400th order zonotopes, each described by 2406 real numbers, to enclose 6-dimensional reachable sets in [64]).

Finally, approaches based on differential inequalities (DI) operate by constructing an auxiliary system of ODEs, twice the size of the original, that describes componentwise upper and lower bounds on the reachable set as its solutions. Harrison [16] originally observed that such a system can be constructed automatically using

simple interval arithmetic. Moreover, this system can be solved with state-of-the-art numerical integration codes, whereas both Taylor series and conservative linearization methods require custom integration algorithms with significant step-size restrictions [34, 5]. Thus, DI methods are capable of producing bounds very rapidly (i.e., at a small multiple of the cost of integrating a single trajectory [64, 59, 61], making DI a potentially powerful tool for real-time control and global dynamic optimization. However, the resulting enclosures are often extremely conservative unless the ODEs satisfy restrictive monotonicity conditions [16]. Several methods have been proposed to address this by enabling the use of more complex reachable set representations in place of intervals. The work [9] proposes an interesting use of DI to compute Taylor model enclosures. However, auxiliary ODEs are required for each Taylor coefficient, which is prohibitive for high-order expansions. The article [77] introduces a general framework for using DI to compute general convex enclosures. Specific implementations can be found in [65, 63, 18]. In particular, the article [18] introduces a very effective method for computing polytopic enclosures using DI. However, this method creates an auxiliary system of ODEs whose right-hand sides are evaluated by solving embedded linear programs rather than using simple IA, which leads to significantly longer computation times.

This chapter presents a new strategy for reducing the conservatism of the DI approach while largely maintaining its efficiency. Rather than using complex non-interval enclosures, the central idea is to exploit model redundancy. This strategy is motivated by very effective DI techniques that have recently been developed for a special class of systems whose solutions are known to satisfy natural bounds (e.g., nonnegativity) and linear relationships (e.g., conservation laws) that are implicitly ensured by, and hence redundant with, the given ODEs [72, 58, 64]. For such systems, DI methods have been developed that exploit these redundant relationships to achieve

much sharper enclosures than standard DI. Moreover, this is accomplished using only fast interval operations, so that the speed of the standard DI method is retained [64]. Motivated by these observations, we present a new approach for arbitrary nonlinear systems based on the deliberate introduction of carefully selected redundant equations that can be exploited within a similar DI bounding procedure. This can be viewed as a dynamic analogue of methods commonly used to generate redundant constraints in global optimization and constraint satisfaction algorithms, such as the reformulation linearization technique (RLT) [70]. Although a fully automated method for selecting redundant equations is not yet available, we demonstrate this strategy through several detailed case studies, which clearly show that redundancy can dramatically reduce conservatism. The additional cost is modest in most cases, but does become significant when many redundant equations are used, highlighting the need for future work on selection heuristics. The mechanisms by which this approach reduces conservatism are discussed in detail, and we provide preconditioning heuristics that significantly improve the efficacy of the added equations. Although we only consider the DI approach here, our results suggest that the addition of redundant equations could be used to effectively reduce conservatism in other approaches as well, potentially enabling the use of lower complexity sets. Indeed, it has already been shown in [76] that pre-existing affine solution invariants can stabilize the enclosures computed by Taylor series methods.

The remainder of this chapter is organized as follows. The formal problem statement is given in Section 2.2. Section 2.3 provides related background on interval analysis and differential inequalities. Section 2.4 presents our new technique. An algorithm for effectively preconditioning redundant linear equations for use in the bounding algorithm is presented in Section 2.5.1. Finally, Section 2.6 presents several case studies.

## 2.2 Problem Statement

Let $I = [t_0, t_f] \subset \mathbb{R}$, let $P \subset \mathbb{R}^{n_p}$ denote a compact set of time-invariant uncertain parameters $\mathbf{p}$, let $D \subset \mathbb{R}^{n_x}$ be open, and let $\mathbf{f} : I \times P \times D \to \mathbb{R}^{n_x}$ and $\mathbf{x}_0 \colon P \to D$ be continuous functions. We consider dynamic processes that can be modeled by systems of nonlinear ODEs of the form

$$\begin{aligned} \dot{\mathbf{x}}(t, \mathbf{p}) &= \mathbf{f}(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p})), \\ \mathbf{x}(t_0, \mathbf{p}) &= \mathbf{x}_0(\mathbf{p}), \end{aligned} \tag{2.1}$$

where a solution is any continuously differentiable mapping $\mathbf{x} : I \times P \to D$ that satisfies (4.1a) for all $(t, \mathbf{p}) \in I \times P$. The following assumption holds throughout.

**Assumption 1** *For any $\mathbf{z} \in D$, there exists $\eta > 0$ and $\alpha \in \mathbb{R}$ such that, for all $t \in I$ and $\mathbf{p} \in P$,*

$$\|\mathbf{f}(t, \mathbf{p}, \tilde{\mathbf{z}}) - \mathbf{f}(t, \mathbf{p}, \hat{\mathbf{z}})\|_\infty \leq \alpha \|\tilde{\mathbf{z}} - \hat{\mathbf{z}}\|_\infty,$$

*for every $\tilde{\mathbf{z}}, \hat{\mathbf{z}} \in B_\eta(\mathbf{z})$, where $B_\eta(\mathbf{z})$ denotes an open ball with radius $\eta$ centered at $\mathbf{z}$.*

Assumption 1 guarantees the local existence and uniqueness of a solution of (4.1a) [10]. In this chapter, it is always assumed that a unique solution of (4.1a) exists on all of $I$, for every $\mathbf{p} \in P$. We are interested in computing an enclosure of these solutions in terms of state bounds, as defined below.

**Definition 2** *Define the* reachable set *of* (4.1a) *at $t \in I$ as*

$$Re(t) \equiv \{\mathbf{x}(t, \mathbf{p}) \colon \mathbf{p} \in P\} \tag{2.2}$$

**Definition 3** *Two functions* $\mathbf{x}^L, \mathbf{x}^U : I \to \mathbb{R}^{n_x}$ *are called* state bounds *if* $\mathbf{x}^L(t) \leq \mathbf{x}(t, \mathbf{p}) \leq \mathbf{x}^U(t)$, $\forall (t, \mathbf{p}) \in I \times P$, *or equivalently,* $Re(t) \subset [\mathbf{x}^L(t), \mathbf{x}^U(t)]$, $\forall t \in I$.

The best possible state bounds describe the interval hull of $Re(t)$, while all others are conservative. Our aim is to develop a method that can exploit model redundancy to compute state bounds with minimal conservatism while maintaining the computational efficiency of standard differential inequalities methods.

## 2.3 Background

### 2.3.1 Interval Arithmetic

This section briefly reviews some concepts from interval arithmetic (IA), which is central to the state bounding methods discussed in later sections. Detailed introductions can be found in [44, 43, 48].

Let the compact $n$-dimensional interval $\left\{\mathbf{x} \in \mathbb{R}^n : \mathbf{x}^L \leq \mathbf{x} \leq \mathbf{x}^U\right\}$ denoted by $X = [\mathbf{x}^L, \mathbf{x}^U]$, and denote the set of all such intervals by $\mathbb{IR}^n$. Similarly, for $D \subset \mathbb{R}^n$, let $\mathbb{I}D$ denote the set of intervals $X$ such that $X \subset D$. An interval $X$ is called degenerate if $\mathbf{x}^L = \mathbf{x}^U$. The midpoint and radius of $X$ are $\mathbf{x}_m = \frac{1}{2}(\mathbf{x}^L + \mathbf{x}^U)$ and $\mathbf{x}_r = \frac{1}{2}(\mathbf{x}^U - \mathbf{x}^L)$, respectively. Thus, $X$ can be also written as $X = \{\mathbf{x}_m + \mathrm{diag}(\mathbf{x}_r)\boldsymbol{\xi} : \boldsymbol{\xi} \in [-\mathbf{1}, \mathbf{1}]\}$, where diag forms a diagonal matrix from its vector argument.

A central task in interval arithmetic is to bound the range of a function over an interval subset of its domain. Theorem 1 below provides a means to accomplish this using so-called *interval extensions*.

**Definition 4** *Let* $D \subset \mathbb{R}^n$ *and* $\mathbf{f} : D \to \mathbb{R}^m$. *An interval function* $F : \mathbb{I}D \to \mathbb{IR}^m$ *is an* interval extension *of* $\mathbf{f}$ *if* $F([\mathbf{x}, \mathbf{x}]) = \mathbf{f}(\mathbf{x})$ *for all degenerate intervals* $[\mathbf{x}, \mathbf{x}] \in \mathbb{I}D$.

**Definition 5** *An interval function* $F\colon \mathbb{I}D \to \mathbb{IR}^m$ *is* inclusion monotonic *if,* $\forall X, Y \in \mathbb{I}D$*,*

$$Y \subset X \Longrightarrow F(Y) \subset F(X). \tag{2.3}$$

**Theorem 1** *If* $F\colon \mathbb{I}D \to \mathbb{IR}^m$ *is an inclusion monotonic interval extensions of* $\mathbf{f}\colon D \to \mathbb{R}^m$*, then [44]*

$$\mathbf{f}(X) \subset F(X), \quad \forall X \in \mathbb{I}D, \tag{2.4}$$

*where* $\mathbf{f}(X)$ *denotes the exact range* $\{\mathbf{f}(\mathbf{x})\colon \mathbf{x} \in X\}$*.*

Computing the exact range $\mathbf{f}(X)$ is generally very difficult. However, by Theorem 1, an interval enclosure of $\mathbf{f}(X)$ can be obtained if an inclusion monotonic interval extension is available. Fortunately, interval arithmetic provides a simple and efficient means to construct such an extension for so-called *factorable functions*.

A function is called factorable if it can be formed by finite recursive composition of basic operations, including the binary operations $\{+, -, \times, \div\}$ and a library of intrinsic univariate functions such as $e^x$, $x^n$, etc. This includes essentially all functions that can be written explicitly in computer code. For example, the function $f(x) = x(1 - e^x)$ is factorable because it can be decomposed into basic operations as

follows:

$$\begin{aligned} v_1 &= x \\ v_2 &= e^{v_1} \\ v_3 &= 1 - v_2 \\ v_4 &= v_1 \times v_3 \\ f &= v_4 \end{aligned} \tag{2.5}$$

Equation (2.5) is called the factorable representation of $f$.

For any factorable function, a particular interval extension called the *natural interval extension* can be constructed by replacing each basic operation in its factorable representation with an interval extension of that operation (these are known and compiled in [44]). For example, the natural interval extension of $f$ is evaluated at $X$ by executing the sequence of computations:

$$\begin{aligned} V_1 &= X \\ V_2 &= e^{V_1} = [e^{v_1^L}, e^{v_1^U}] \\ V_3 &= 1 - V_2 = [1 - v_2^U, 1 - v_2^L] \\ V_4 &= V_1 \times V_3 \\ &= [\min M, \max M], \;\; M = \{v_1^L v_3^L, v_1^U v_3^U, v_1^L v_3^U, v_1^U v_3^L\} \\ F &= V_4 \end{aligned} \tag{2.6}$$

### 2.3.2 Origins of conservatism in Interval Arithmetic

Using natural interval extensions, an enclosure of the range of any factorable function over an interval can be computed automatically and very efficiently. How-

ever, the resulting enclosure is often conservative for one of two reasons. These are briefly described here because they motivate the method presented in §2.4 for reducing the conservatism of state bounding methods.

*The dependency problem* refers to the fact that the binary interval operations $\{+, -, \times, \div\}$ always treat their operands as independent. For example, in (2.5), the variables $v_1 = x$ and $v_3 = 1 - e^x$ are dependent because they are related through $x$, but the interval extension of the product $v_1 v_3$ in (2.6) conservatively assumes that $v_1$ and $v_3$ can vary independently within $V_1$ and $V_3$, respectively. With $x \in [-1, 1]$, this gives $[1 - e, e - 1]$, which overestimates the true range $[1 - e, 0]$. Dependency causes overestimation whenever the same variable appears multiple times in an expression, and it often cannot be addressed by simple algebraic rearrangements.

*The wrapping effect* refers to the conservatism introduced by using an interval to enclose (or *wrap*) a non-interval set. This term is typically used only in the context of bounding dynamic systems, but it can be understood more simply by considering the interval extension of a vector function such as $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x})) = (x_1, x_1 + x_2)$. The range of $\mathbf{f}$ with $x_1, x_2 \in [0, 1]$ is a line segment from $(0, 0)$ to $(1, 2)$, while IA gives $[0, 1] \times [0, 2]$. The problem arises because IA treats the appearance of $x_1$ in $f_1$ as independent of the appearance of $x_1$ in $f_2$. Thus, in this context, the wrapping affect can be considered as another form of dependency, although neither $f_1$ or $f_2$ suffer from dependency themselves.

Two strategies are typically used to mitigate the conservatism caused by dependency and wrapping. The first is to partition $X \subset \mathbb{R}^n$ into subintervals and bound $f$ on each one separately. Since the number of subintervals grows exponentially in $n$, partitioning alone is often an unsatisfactory solution. The second approach is to use arithmetics based on more complex, non-interval sets. This can greatly mitigate dependency and wrapping, but often at significant cost [38]. In this chapter, our interest

is in a third approach, the use of redundancy, which is based on the observation that, because of dependency, equivalent expressions for a function $\mathbf{f}(\mathbf{x})$ can result in distinct enclosures. For example, let $f(x) = x^2 - x$ and $g(x) = x(x-1)$. With $x \in [0,1]$, IA gives $f(x) \in [0,1]^2 - [0,1] = [-1,1]$ and $g(x) \in [0,1]([0,1]-1) = [0,1][-1,0] = [-1,0]$. Thus, the original definition of $f$ can be augmented with the redundant equation $f = g$ to deduce the improved bound $f(x) \in [-1,1] \cap [-1,0]$. In general, one enclosure need not be a subset of the other.

Redundancy is extensively used to improve bounding procedures used in constraint satisfaction and global optimization codes for *non-dynamic* problems [40, 27, 11, 79, 70]. Experience in these applications has shown that a small number of carefully chosen redundant constraints can dramatically reduce conservatism at minor additional cost, although identifying these constraints is challenging.

### 2.3.3 The Standard Differential Inequalities Method

This section presents the standard differential inequalities (DI) method for computing state bounds for the ODEs (4.1a), which is central to the new methods developed in §2.4. The DI method is based on Theorem 2 below, which provides sufficient conditions for two trajectories to be state bounds. See [64] for proof.

**Definition 6** *For each* $i \in \{1, \ldots, n_x\}$, *define* $\mathcal{B}_i^L, \mathcal{B}_i^U : \mathbb{IR}^{n_x} \to \mathbb{IR}^{n_x}$ *by* $\mathcal{B}_i^L([\mathbf{x}^L, \mathbf{x}^U]) = \{\mathbf{z} \in [\mathbf{x}^L, \mathbf{x}^U] : z_i = x_i^L\}$ *and* $\mathcal{B}_i^U([\mathbf{x}^L, \mathbf{x}^U]) = \{\mathbf{z} \in [\mathbf{x}^L, \mathbf{x}^U] : z_i = x_i^U\}$ .

**Theorem 2** *Let* $\mathbf{x}^L, \mathbf{x}^U : I \to \mathbb{R}^{n_x}$ *be continuous functions, define* $X(t) \equiv [\mathbf{x}^L(t), \mathbf{x}^U(t)]$, *and assume:*

1. *For every* $t \in I$ *and every index* $i$,

   (a) $\mathbf{x}^L(t) \leq \mathbf{x}^U(t)$,

*(b)* $\mathcal{B}_i^L(X(t)) \subset D$, $\mathcal{B}_i^U(X(t)) \subset D$,

*2.* $\mathbf{x}_0(\mathbf{p}) \in X(t_0), \forall \mathbf{p} \in P$.

*3. For all* $t \in I$ *and each index* $i$,

*(a)* $\dot{x}_i^L(t) \leq f_i(t, \mathbf{p}, \mathbf{z})$, $\forall (\mathbf{p}, \mathbf{z}) \in P \times \mathcal{B}_i^L(X(t))$,

*(b)* $\dot{x}_i^U(t) \geq f_i(t, \mathbf{p}, \mathbf{z})$, $\forall (\mathbf{p}, \mathbf{z}) \in P \times \mathcal{B}_i^U(X(t))$.

*Then* $\mathbf{x}(t, \mathbf{p}) \in X(t)$, $\forall (t, \mathbf{p}) \in I \times P$.

Hypotheses 2 and 3 are the key requirements in Theorem 2. Hypothesis 2 requires that $\mathbf{x}^L(t)$ and $\mathbf{x}^U(t)$ bound all solutions of (4.1a) at $t = t_0$, while Hypothesis 3 ensures that this property is maintained to the right of $t_0$ by requiring that $\mathbf{x}^L(t)$ and $\mathbf{x}^U(t)$ decrease and increase, respectively, faster than any solution of (4.1a). The interval functions $\mathcal{B}_i^{L/U}$ arise in 3(a) and 3(b) because, theoretically, it is only necessary for $x_i^L(t)$ to decrease faster than the $i^{\text{th}}$ component of trajectories $\mathbf{x}(t, \mathbf{p})$ that are already incident on the $i^{\text{th}}$ lower bound (i.e., $x_i(t, \mathbf{p}) = x_i^L(t)$), and similarly for $x_i^U(t)$. Note that both Hypotheses 2 and 3 concern the range of the functions $x_{0,i}$ and $f_i$ over interval subsets of their domains. Assuming that $x_{0,i}$ and $f_i$ are factorable, let $X_{0,i}(P) = [x_{0,i}^L(P), x_{0,i}^U(P)]$ and $F_i([t,t], P, Z) = [f_i^L([t,t], P, Z), f_i^U([t,t], P, Z)]$ denote their natural interval extensions. Then, as originally proposed in [16], state bounds satisfying the conditions of Theorem 2 can be computed as the solutions of following auxiliary systems of $2n_x$ ODEs:

$$\begin{aligned} \dot{x}_i^L(t) &= f_i^L([t,t], P, \mathcal{B}_i^L([\mathbf{x}^L(t), \mathbf{x}^U(t)])), \\ \dot{x}_i^U(t) &= f_i^U([t,t], P, \mathcal{B}_i^U([\mathbf{x}^L(t), \mathbf{x}^U(t)])), \\ [x_i^L(t_0), x_i^U(t_0)] &= X_{i,0}(P), \end{aligned} \tag{2.7}$$

for all $t \in I$ and $i \in \{1, \ldots, n_x\}$. This system can be solved using any state-of-the-art numerical integrator to produce state bounds very efficiently, making DI a potentially powerful tool for real-time applications.

### 2.3.4 Origins of Conservatism in Differential Inequalities

Despite its speed, DI often produces state bounds that are far too conservative to be useful in applications [16]. This section outlines the main causes of this conservatism in order to motivate our new approach for mitigating it in §2.4. The first cause is simply that the required interval extensions of each $f_i$ in (2.7) often suffer from the dependency problem as described in §2.3.2. However, there are more subtle problems that are not related to the shortcomings of IA, but rather arise from conservatism in the conditions of Theorem 2 itself. To see this, consider the ODEs

$$\begin{aligned} \dot{x}_1(t, \mathbf{p}) &= -x_2(t, \mathbf{p}), \\ \dot{x}_2(t, \mathbf{p}) &= x_1(t, \mathbf{p}), \end{aligned} \tag{2.8}$$

with $x_1(t_0, \mathbf{p}) = p_1 \in [0, 1]$ and $x_2(t_0, \mathbf{p}) = p_2 \in [0, 1]$. The right-hand side functions in (2.8) do not have dependency problems in the sense of §2.3.2 (i.e., multiple instances of the same variable). It follows that the interval extensions in (2.7) will coincide with the exact ranges of these functions, and hence Hypotheses 3(a)–(b) of Theorem 2 will be satisfied with equality. Nevertheless, DI produces very conservative bounds for this system. A simple explanation is that the reachable set is not an interval for most $t$, so the state bounds suffer from the wrapping effect as shown in Figure 2.1. A more illuminating explanation is that, for any $t > t_0$, the variables $x_1$ and $x_2$ are dependent (both depend on $\mathbf{p}$). We call this *historical dependency*. Given this fact, consider computing a lower bound on one of the right-hand side functions $f_i$ in (2.8) over the

set $\{\mathbf{z} \in [\mathbf{x}^L(t), \mathbf{x}^U(t)] : z_i = x_i^L(t)\}$, as required by Hypothesis 3(a) of Theorem 2 (a similar argument holds for 3(b)). The purpose of this condition is to ensure that $\dot{x}_i^L(t)$ is lower than any value of $\dot{x}_i(t, \mathbf{p}) = f_i(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p}))$ achievable by a real solution of (2.8) that is already incident on the $i^{\text{th}}$ lower bound, if any. In other words, $\dot{x}_i^L(t)$ must be lower than $f_i(t, \mathbf{p}, \mathbf{z})$ for all $\mathbf{z}$ in the set $\{\mathbf{z} \in Re(t) : z_i = x_i^L(t)\}$. But historical dependency implies that this set is overestimated by $\{\mathbf{z} \in [\mathbf{x}^L(t), \mathbf{x}^U(t)] : z_i = x_i^L(t)\}$. Given the discussion of the dependency problem in §2.3.2, it is reasonable to suspect that this will only be problematic if $f_i$ depends on both $x_1$ and $x_2$, since such an $f_i$ would then be an expression containing two (historically) dependent variables that are required to be treated independently in the bounding procedure. Remarkably, this is not even necessary for conservatism to arise. For example, the right-hand side function $f_1(t, \mathbf{z}) = -z_2$ in (2.8) is required by Theorem 2 to be bounded over $\{\mathbf{z} \in [\mathbf{x}^L(t), \mathbf{x}^U(t)] : z_1 = x_1^L(t)\}$, which allows $z_2$ to take any value in $[x_2^L(t), x_2^U(t)]$. However, Figure 2.1 shows that (e.g., at $t = 2$) the set $\{\mathbf{z} \in Re(t) : z_1 = x_1^L(t)\}$ is a singleton if $[\mathbf{x}^L(t), \mathbf{x}^U(t)]$ is the interval hull of $Re(t)$, and is empty otherwise.

However it arises for a given system of ODEs, conservatism in DI tends to grow rapidly because overestimation in $[\mathbf{x}^L(t), \mathbf{x}^U(t)]$ at $t$ enlarges the feasible sets in Hypotheses 3(a)–(b), which in turn affects all $[\mathbf{x}^L(s), \mathbf{x}^U(s)]$ with $s > t$. This phenomenon is similar to the well-known exponential growth of numerical errors for unstable systems, and often results exponential divergence of the bounds.

### 2.3.5 State Bounds using a priori Enclosures

This section briefly describes a very effective strategy for reducing the conservatism of DI for a special class of systems, originally developed in [64], and in earlier forms in [58, 72]. The new DI methods presented in this chapter rely on the key

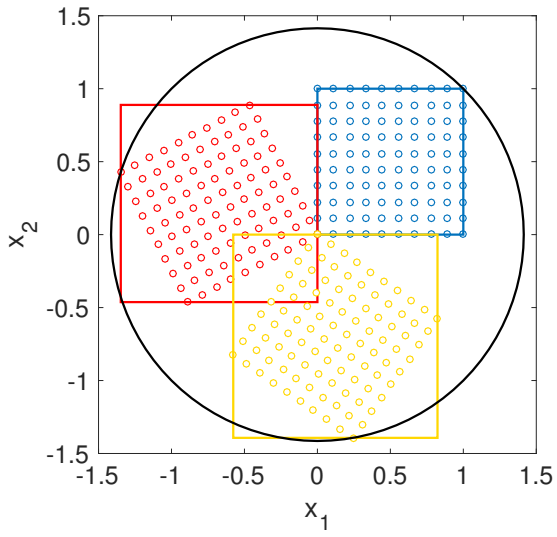


Figure 2.1: Real solutions $\mathbf{x}(t, \mathbf{p})$ of (2.8) (circles) and the interval hull of $Re(t)$ (solid line) at $t = 0$ (blue), $t = 2$ (red), and $t = 4$ (yellow). The solid circle is an *a priori* enclosure as discussed in §2.3.5.

insights of this approach.

The method in [64] applies to ODEs that are known *a priori* to satisfy some crude enclosure $Re(t) \subset G$, $\forall t \in I$. The set $G$ may represent physical state bounds such as nonnegativity, or more complex relations such as conservation laws. For example, it is easy to verify that all solutions of (2.8) satisfy $\|\mathbf{x}(t, \mathbf{p})\|_2^2 = \|\mathbf{x}(t_0, \mathbf{p})\|_2^2$, $\forall (t, \mathbf{p}) \in I \times P$. Moreover, it is clear from Figure 2.1 that this constraint might be useful for bounding since even the interval hull of $Re(t)$ encloses points that violate it.

The central result of [64] states that an *a prior* enclosure can be used to weaken the requirements of Theorem 2 and thereby enable the efficient computation of much more accurate bounds. The key idea is to enforce the inequality in, e.g., Hypothesis 3(a), for only those $\mathbf{z} \in \mathcal{B}_i^L([\mathbf{x}^L(t), \mathbf{x}^U(t)]) \cap G$, rather than for all $\mathbf{z} \in \mathcal{B}_i^L([\mathbf{x}^L(t), \mathbf{x}^U(t)])$. This potentially combats the more subtle sources of conservatism discussed in the previous section. Specifically, if $G$ is not an interval, than it contains

at least some information about the historical dependency of the states $\mathbf{x}(t, \mathbf{p})$ on one another, and makes this information available when bounding the range of each $f_i$. Although this modification of Hypothesis 3(a) is not valid exactly as written above, it holds if $G$ is enforced through an interval refinement procedure $\mathcal{I}_G$ satisfying the following requirements.

**Assumption 2** *Let $\mathcal{I}_G : \mathbb{IR}^{n_x} \to \mathbb{IR}^{n_x}$ satisfy*

1. *$\mathcal{I}_G(Z) \subset Z$ for all $Z \in \mathbb{IR}^{n_x}$ with $Z \cap G \neq \emptyset$,*
2. *$\forall Z \in \mathbb{IR}^{n_x}$, if $\mathbf{z} \in Z$ and $\mathbf{z} \notin \mathcal{I}_G(Z)$, then $\mathbf{z} \notin G$,*
3. *$\exists L_{\mathcal{I}} \in \mathbb{R}_+$ such that, $\forall Z_1, Z_2 \in \mathbb{IR}^{n_x}$,*

$$d_H(\mathcal{I}_G(Z_1), \mathcal{I}_G(Z_2)) \leq L_{\mathcal{I}} d_H(Z_1, Z_2), \tag{2.9}$$

*where $d_H$ denotes the Hausdorff metric.*

**Theorem 3** *Let $\mathbf{x}^L, \mathbf{x}^U : I \to \mathbb{R}^{n_x}$ be continuous functions, define $X(t) \equiv [\mathbf{x}^L(t), \mathbf{x}^U(t)]$, and assume:*

1. *For every $t \in I$ and every index $i$,*
   (a) *$\mathbf{x}^L(t) \leq \mathbf{x}^U(t)$,*
   (b) *$\mathcal{I}_G(\mathcal{B}_i^L(X(t))) \subset D$, $\mathcal{I}_G(\mathcal{B}_i^U(X(t))) \subset D$.*
2. *$\mathbf{x}_0(\mathbf{p}) \in X(t_0), \forall \mathbf{p} \in P$.*
3. *For all $t \in I$ and each index $i$,*
   (a) *$\dot{x}_i^L(t) \leq f_i(t, \mathbf{p}, \mathbf{z})$, $\forall (\mathbf{p}, \mathbf{z}) \in P \times \mathcal{I}_G(\mathcal{B}_i^L(X(t)))$,*
   (b) *$\dot{x}_i^U(t) \geq f_i(t, \mathbf{p}, \mathbf{z})$, $\forall (\mathbf{p}, \mathbf{z}) \in P \times \mathcal{I}_G(\mathcal{B}_i^U(X(t)))$.*

*Then* $\mathbf{x}(t,\mathbf{p}) \in X(t)$, $\forall (t,\mathbf{p}) \in I \times P$.

By analogy to (2.7), state bounds satisfying the requirements of Theorem 3 can be computed as the solutions of

$$\begin{aligned} \dot{x}_i^L(t) &= f_i^L([t,t], P, \mathcal{I}_G(\mathcal{B}_i^L([\mathbf{x}^L(t), \mathbf{x}^U(t)]))), \\ \dot{x}_i^U(t) &= f_i^U([t,t], P, \mathcal{I}_G(\mathcal{B}_i^U([\mathbf{x}^L(t), \mathbf{x}^U(t)]))), \\ [x_i^L(t_0), x_i^U(t_0)] &= X_{i,0}(P), \end{aligned} \tag{2.10}$$

for all $t \in I$ and each index $i$. Solving (2.10) often produces much more accurate state bounds than (2.7), and requires only moderately more effort provided that $\mathcal{I}_G$ can be evaluated using fast interval computations. In [64], an efficient $\mathcal{I}_G$ was defined for the special case of polyhedral *a priori* enclosures $G = \{\mathbf{z} \in X_{\text{nat}} : \mathbf{M}\mathbf{z} \leq \mathbf{b}\}$, where $\mathbf{M} \in \mathbb{R}^{n_m \times n_x}$, $\mathbf{b} \in \mathbb{R}^{n_m}$, and $X_{\text{nat}}$ is a possibly unbounded interval of *natural bounds* (i.e., nonnegativity). For reference in §2.5.2, this definition is given in Algorithm 1, modified for consistency with §2.5.2 to apply more specifically to enclosures of the form $G = \{\mathbf{z} \in X_{\text{nat}} : \mathbf{M}\mathbf{z} = \mathbf{b}\}$. The idea is to consider, for each $m_{ij} \neq 0$, the rearrangement of the $i^{\text{th}}$ equation for $z_j$,

$$z_j = m_{ij}^{-1}(b_i - \textstyle\sum_{k \neq j} m_{ik} z_k). \tag{2.11}$$

Given an initial interval bound $Z$, $Z_j$ can potentially be refined by bounding the right-hand side of (2.11) using IA. $\mathcal{I}_G$ is defined by applying this refinement to every possible choice of $m_{ij} \neq 0$. In Algorithm 1, $\text{mid}(a,b,c)$ returns the middle value of $a$, $b$, and $c$, and is used in place of $\max(z_j^L, \zeta)$ and $\min(z_j^U, \gamma)$ in lines 8 and 9, respectively, to avoid returning an empty interval when $[\mathbf{z}^L, \mathbf{z}^U] \cap G = \emptyset$.

Of course, the drawback of computing state bounds via (2.10) is that it only

**Algorithm 1** $\mathcal{I}_G$ defined in [64]

**function** IG($\mathbf{z}^L, \mathbf{z}^U, X_{\text{nat}}, \mathbf{M}, \mathbf{b}, \text{tol}$)
  $[\mathbf{z}^L, \mathbf{z}^U] \leftarrow [\mathbf{z}^L, \mathbf{z}^U] \cap X_{\text{nat}}$
  **for** $j \leftarrow 1, n_x$ **do**
    **for** $i \leftarrow 1, n_m$ **do**
      **if** $|m_{ij}| > \text{tol}$ **then**
        $\zeta \leftarrow \frac{b_i}{m_{ij}} + \sum_{k \neq j} \min(-\frac{m_{ik}}{m_{ij}} z_k^L, -\frac{m_{ik}}{m_{ij}} z_k^U)$
        $\gamma \leftarrow \frac{b_i}{m_{ij}} + \sum_{k \neq j} \max(-\frac{m_{ik}}{m_{ij}} z_k^L, -\frac{m_{ik}}{m_{ij}} z_k^U)$
        $z_j^L \leftarrow \text{mid}(z_j^L, z_j^U, \zeta)$
        $z_j^U \leftarrow \text{mid}(z_j^L, z_j^U, \gamma)$
      **end if**
    **end for**
  **end for**
  **return** $[\mathbf{z}^L, \mathbf{z}^U]$
**end function**

applies to systems where $G$ is readily available, and offers no means to address the conservatism of DI when this is not the case. Moreover, even when some $G$ is available, this approach does not provide a means to achieve any further improvements once $G$ has been exploited through (2.10).

In order to extend this approach in the next section, it is important to note that valid $G$ sets are implied by, and hence *redundant* with, the given ODEs. For example, the set depicted in Figure 2.1 is easy derived from (2.8) by simply verifying that $\frac{d}{dt}\|\mathbf{x}(t, \mathbf{p})\|_2^2 = 0$. Thus, although it seems like the approach above is introducing new information into the bounding procedure, it is not. However, information that is redundant in real arithmetic is not necessarily redundant in interval arithmetic (see §2.3.2), and this observation extends to DI (see §2.3.4). Thus, the approach above should be understood more precisely as method for using redundant information to partially enforce historical dependency in the interval extensions required in (2.10).

## 2.4 State Bounds using Manufactured Model Redundancy

In this section, we present a new method for computing accurate state bounds that extends the method discussed in §2.3.5 to arbitrary nonlinear ODEs of the form (4.1a). In other words, we do not require any prior knowledge of a set $G$ containting the reachable set. The key idea is to manufacture such a set by deliberately introducing new state variables and ODEs that are redundant with the original states by definition. Specifically, this is done by choosing a continuously differentiable function $\mathbf{g} : D \to \mathbb{R}^{n_y}$, defining the new state variables $\mathbf{y}(t, \mathbf{p}) \equiv \mathbf{g}(\mathbf{x}(t, \mathbf{p}))$, and finally differentiating this definition to form the *augmented system*

$$\begin{aligned} \frac{d}{dt}\begin{bmatrix} \mathbf{x}(t,\mathbf{p}) \\ \mathbf{y}(t,\mathbf{p}) \end{bmatrix} &= \begin{bmatrix} \mathbf{f}(t,\mathbf{p},\mathbf{x}(t,\mathbf{p})) \\ \frac{\partial \mathbf{g}}{\partial \mathbf{x}}(\mathbf{x}(t,\mathbf{p}))\mathbf{f}(t,\mathbf{p},\mathbf{x}(t,\mathbf{p})) \end{bmatrix}, \\ \begin{bmatrix} \mathbf{x}(t_0,\mathbf{p}) \\ \mathbf{y}(t_0,\mathbf{p}) \end{bmatrix} &= \begin{bmatrix} \mathbf{x}_0(\mathbf{p}) \\ \mathbf{g}(\mathbf{x}_0(\mathbf{p})) \end{bmatrix}. \end{aligned} \tag{2.12}$$

Clearly, if $(\mathbf{x}, \mathbf{y}) : I \times P \to \mathbb{R}^{n_x} \times \mathbb{R}^{n_y}$ is a solution of (2.12), then $\mathbf{x}$ is a solution of (4.1a). Thus, state bounds for (2.12) provide state bounds for (4.1a). Moreover, by design, (2.12) implies that

$$\mathbf{y}(t_0, \mathbf{p}) - \mathbf{g}(\mathbf{x}(t_0, \mathbf{p})) = \mathbf{0} \quad \text{and} \tag{2.13}$$

$$\begin{aligned} \frac{d}{dt}[\mathbf{y}(t,\mathbf{p}) - \mathbf{g}(\mathbf{x}(t,\mathbf{p}))] &= \dot{\mathbf{y}}(t,\mathbf{p}) - \frac{\partial \mathbf{g}}{\partial \mathbf{x}}(\mathbf{x}(t,\mathbf{p}))\dot{\mathbf{x}}(t,\mathbf{p}) \\ &= \mathbf{0}, \end{aligned} \tag{2.14}$$

which together imply that the solutions of (2.12) satisfy the invariants $\mathbf{y}(t,\mathbf{p}) - \mathbf{g}(\mathbf{x}(t,\mathbf{p})) = \mathbf{0}$, $\forall (t,\mathbf{p}) \in I \times P$. Equivalently, we have arranged that

$$(\mathbf{x}(t,\mathbf{p}), \mathbf{y}(t,\mathbf{p})) \in G \equiv \{(\mathbf{z}_x, \mathbf{z}_y) : \mathbf{z}_y = \mathbf{g}(\mathbf{z}_x)\}, \tag{2.15}$$

for all $(t,\mathbf{p}) \in I \times P$. Thus, state bounds for (2.12) can be computed using the method described in §2.3.5 as follows.

**Theorem 4** *Choose any differentiable* $\mathbf{g} : D \to \mathbb{R}^{n_y}$ *such that* $\frac{\partial \mathbf{g}}{\partial \mathbf{x}}$ *is locally Lipschitz continuous on* $D$ *and* (2.12) *has a unique solution on* $I$ *for every* $\mathbf{p} \in P$. *Define* $G$ *as in* (2.15) *and let* $\mathcal{I}_G$ *satisfy Assumption 2. Moreover, let*

$$\mathbf{h}(t,\mathbf{p},\mathbf{z}) \equiv \frac{\partial \mathbf{g}}{\partial \mathbf{x}}(\mathbf{z})\mathbf{f}(t,\mathbf{p},\mathbf{z}), \tag{2.16}$$

*for all* $(t,\mathbf{p},\mathbf{z}) \in I \times P \times D$, *and let* $G_j$ *and* $[h_j^L, h_j^U]$ *be inclusion monotonic interval extensions of* $g_j$ *and* $h_j$, *respectively. Finally, let* $\mathbf{x}^L, \mathbf{x}^U : I \to \mathbb{R}^{n_x}$ *and* $\mathbf{y}^L, \mathbf{y}^U : I \to \mathbb{R}^{n_y}$ *be solutions of the ODEs*

$$\begin{aligned}
\dot{x}_i^L(t) &= f_i^L([t,t], P, \mathcal{I}_G(\mathcal{B}_i^L(Z(t)))), \\
\dot{x}_i^U(t) &= f_i^U([t,t], P, \mathcal{I}_G(\mathcal{B}_i^U(Z(t)))), \\
\dot{y}_j^L(t) &= h_j^L([t,t], P, \mathcal{I}_G(\mathcal{B}_{n_x+j}^L(Z(t)))), \\
\dot{y}_j^U(t) &= h_j^U([t,t], P, \mathcal{I}_G(\mathcal{B}_{n_x+j}^U(Z(t)))), \\
[x_i^L(t_0), x_i^U(t_0)] &= X_{i,0}(P), \\
[y_j^L(t_0), y_j^U(t_0)] &= G_j(X_0(P)),
\end{aligned} \tag{2.17}$$

*for all* $i \in \{1,\ldots,n_x\}$ *and* $j \in \{1,\ldots,n_y\}$, *where* $Z(t) = \left[\begin{bmatrix} \mathbf{x}^L(t) \\ \mathbf{y}^L(t) \end{bmatrix}, \begin{bmatrix} \mathbf{x}^U(t) \\ \mathbf{y}^U(t) \end{bmatrix}\right]$. *Then* $\mathbf{x}(t,\mathbf{p}) \in [\mathbf{x}^L(t), \mathbf{x}^U(t)]$ *and* $\mathbf{y}(t,\mathbf{p}) = \mathbf{g}(\mathbf{x}(t,\mathbf{p})) \in [\mathbf{y}^L(t), \mathbf{y}^U(t)]$ *for all* $(t,\mathbf{p}) \in I \times P$.

**Proof** The assumptions on $\mathbf{g}$ ensure that (2.12) satisfies Assumption 1. Given (2.15), the result is a direct application of Theorem 3. □

□

Theorem 4 shows that valid state bounds for (4.1a) can be computed by bounding the augmented system (2.12) rather than the original ODEs, and that the manufactured *a prior* enclosure (2.15) can be exploited in doing so. However, Theorem 4 does not ensure that this will result in improved bounds, and provides no insight into the mechanisms through which improvement is possible. Understanding these mechanisms is important because they provide useful information about how to choose effective $\mathbf{g}$ functions. The following example shows that there are indeed choices of $\mathbf{g}$ that produce bounds that are significantly sharper than those produced by applying standard DI to (4.1a). Moreover, it illustrates all of the mechanisms by which improvement is possible, and demonstrates that these depend not only on $\mathbf{g}$, but also on the specific way in which the right-hand sides of (2.12) are expressed. In particular, since expressions that are equivalent in real arithmetic are not always equivalent in interval arithmetic, algebraic rearrangements of the functions $h_j$ can be advantageous. Moreover, the introduction of the new state variables $y_j$ may also permit the functions $f_i$ to be rearranged in a beneficial way, e.g., by writing $f_i(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p})) = \hat{f}_i(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p}))$ for some $\hat{f}_i$.

**Example 1** *Consider the ODEs*

$$\begin{aligned} \dot{x}_1 &= -x_1 x_2 + x_3, \\ \dot{x}_2 &= 2x_1 x_2, \\ \dot{x}_3 &= x_1 + x_2, \end{aligned} \tag{2.18}$$

with $\mathbf{x}(t_0, \mathbf{p}) = (p_1, p_2, p_3) \in [0,1] \times [0,1] \times [0,1]$. To apply Theorem 4, we define a single redundant state $y = x_1 + x_2$. Bounds can now be produced through any of the following four methods.

- *Method* 1: Directly apply standard DI to (2.18).
- *Method* 2: Form an augmented system by adding to (2.18) the new ODE $\dot{y} = \dot{x}_1 + \dot{x}_2 = f_1 + f_2$, but do not make any algebraic rearrangements to the resulting right-hand side function; i.e.,

$$\dot{y} = -x_1 x_2 + x_3 + 2x_1 x_2. \tag{2.19}$$

  The solution of this system satisfies (2.15) with

$$G \equiv \{(\mathbf{z}_x, z_y) : z_y = z_{x,1} + z_{x,2}\}. \tag{2.20}$$

  Using the $\mathcal{I}_G$ function defined for affine solution invariants of this type in [64], compute state bounds via Theorem 4.
- *Method* 3: Simplify the ODE for $y$ as follows and proceed as in Method 2:

$$\dot{y} = x_1 x_2 + x_3. \tag{2.21}$$

- *Method* 4: Simplify the ODE for $x_3$ as follows and proceed as in Method 3:

$$\dot{x}_3 = y. \tag{2.22}$$

Figure 2.2 shows the bounds on $x_3$ produced by each of these four methods, which consistently improve from Method 1 to 4. Each of these improvements results

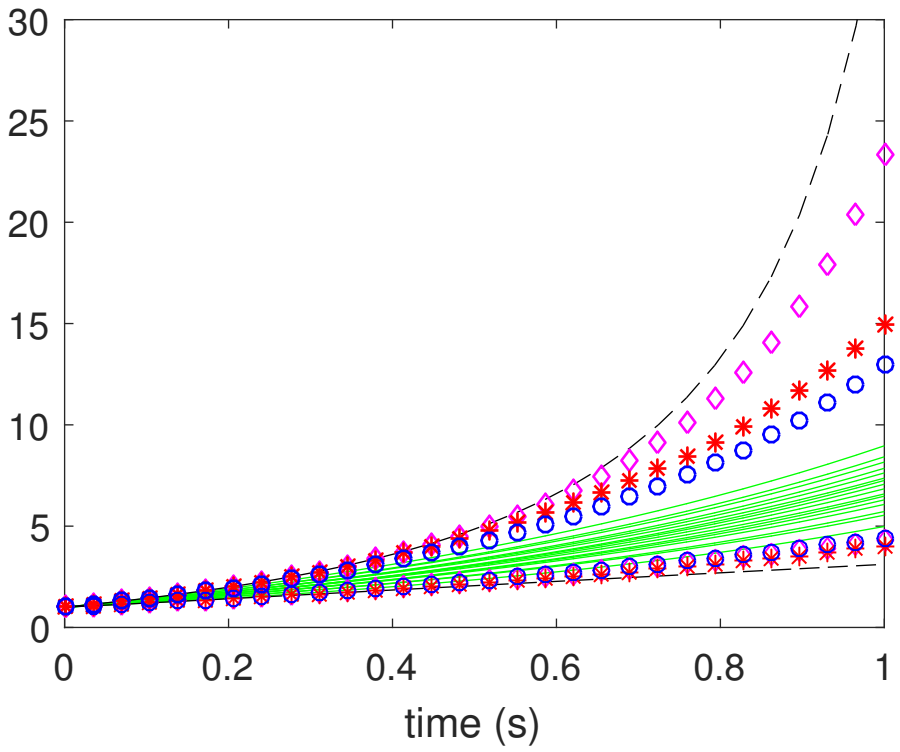


Figure 2.2: Solutions (solid) and state bounds for $x_3$ in (2.18) computed using Methods 1 (dashed), 2 (diamonds), 3 (stars), and 4 (circles).

from at least one of the following three mechanisms.

*New Variable Flattening.* To explain the improvement from Method 1 to 2, first note that the right-hand side for $\dot{x}_3$ depends on $x_1$ and $x_2$. Thus, if the invariant $y = x_1 + x_2$ can be used to improve the bounds on $x_1$ and $x_2$ at some $t \in I$ (i.e., if $\mathcal{I}_G(\mathcal{B}_3^{L/U}(Z(t)))$ is a strict subset of $\mathcal{B}_3^{L/U}(Z(t)))$, then the bounds on $x_3$ will improve to the right of $t$. However, for the invariant to be effective in this way at $t$, at least one bound of $[y^L(t), y^U(t)]$ must be tighter than the corresponding bound of $[x_1^L(t) + x_2^L(t), x_1^U(t) + x_2^U(t)]$. Otherwise, the relation $y = x_1 + x_2$ provides no information at $t$ that is not already available from the bounds on $x_1$ and $x_2$. In this example, it happens that $[y^L(t), y^U(t)]$ is a strict subset of $[x_1^L(t) + x_2^L(t), x_1^U(t) + x_2^U(t)]$ for all $t > t_0$, leading to the observed improvement in the bounds for $x_3$.

However, this leaves the question of how $[y^L(t), y^U(t)]$ came to be sharper than $[x_1^L(t)+x_2^L(t), x_1^U(t)+x_2^U(t)]$ in the first place. By definition, these intervals are equal at $t_0$. Moreover, because $\dot{y}$ is specified directly as $f_1+f_2$ with no algebraic simplifications, it is reasonable to suspect that, e.g., $\dot{y}^L(t) = \dot{x}_1^L + \dot{x}_2^L$ for all $t > t_0$, so that these intervals will remain equal for all $t > t_0$. However, note that in (2.17), $f_1$ and $f_2$ are bounded over $\mathcal{I}_G(\mathcal{B}_1^L(Z(t)))$ and $\mathcal{I}_G(\mathcal{B}_2^L(Z(t)))$, respectively, when evaluating $\dot{x}_1^L(t)$ and $\dot{x}_2^L(t)$, whereas they are bounded over $\mathcal{I}_G(\mathcal{B}_4^L(Z(t)))$ when evaluating $\dot{y}^L(t)$. Note that the set $\mathcal{I}_G(\mathcal{B}_4^L(Z(t)))$ is an interval enclosure of

$$\mathcal{B}_4^L(Z(t)) \cap G = \{(x_1, x_2, x_3, y) \in Z(t) : x_1 + x_2 = y = y^L(t)\}. \tag{2.23}$$

This 'flattening' of $y$, and hence of $x_1 + x_2$, to its lower bound is a new feature of Method 2 in the sense that no such set is ever considered in standard DI (i.e., Method 1). For this particular example, it happens that $\mathcal{I}_G(\mathcal{B}_4^L(Z(t)))$ is a singleton at $t_0$, as shown in Figure 2.3, and 'bounding' $f_1+f_2$ over this set as required by (2.17) actually

leads to $\dot{y}^L(t_0) > \dot{x}_1^L(t_0) + \dot{x}_2^L(t_0)$, and hence $y^L(t) > x_1^L(t) + x_2^L(t)$ immediately to the right of $t_0$. At later times analysis becomes difficult, but numerical results show that $[y^L(t), y^U(t)]$ remains sharper than $[x_1^L(t) + x_2^L(t), x_1^U(t) + x_2^U(t)]$ for all $t \in I$.

*Algebraic simplification of $h_i$.* The improvement from Method 2 to 3 results from the term cancellation carried out in the right-hand side function for $\dot{y}$. In Method 2, this right-hand side function suffers from the dependency problem as described in §2.3.2 because there are multiple appearances of both $x_1$ and $x_2$. In Method 3, the dependency problem is eliminated. Thus, the interval extensions of this function in (2.17) will provide a more accurate enclosure of its range than is achieved in Method 2 (in fact, it is exact in this case). Compared with the affects of *new variable flattening* discussed above, this improvement makes $[y^L(t), y^U(t)]$ even tighter relative to $[x_1^L(t) + x_2^L(t), x_1^U(t) + x_2^U(t)]$. Thus, the invariant $y = x_1 + x_2$ can be used even more effectively to refine the bounds on $x_1$ and $x_2$ at every $t \in I$, which in turn affects $x_3$ through better bounding of its right-hand side function.

*Substitution of y into the right-hand sides.* The improvement from Method 3 to 4 is achieved because the interval $[y^L(t), y^U(t)]$ is sharper than $[x_1^L(t) + x_2^L(t), x_1^U(t) + x_2^U(t)]$ for all $t \in I$, as discussed above in regards to Methods 2 and 3. Thus, the bounding system (2.17) specifies upper and lower bounds on $x_3$ that increase and decrease, respectively, less in Method 4 than in Method 3. This difference can be eliminated if, in Method 3, the refinement operation $\mathcal{I}_G$ is modified from the definition given in [64]. Specifically, the invariant $y = x_1 + x_2$ must be used to infer that the larger interval $[x_1^L(t) + x_2^L(t), x_1^U(t) + x_2^U(t)]$ can be replaced by the smaller interval $[y^L(t), y^U(t)]$ when bounding the right-hand side for $\dot{x}_3$. However, the given definition of $\mathcal{I}_G$ uses the invariant to improve the bounds on individual states if possible, not on combinations of them. Specifically, in this case it aims to improve the bounds on

$x_1$ through the rearrangement

$$\begin{aligned} x_1(t,\mathbf{p}) &= y(t,\mathbf{p}) - x_2(t,\mathbf{p}) \\ &\in [y^L(t) - x_2^U(t), y^U(t) - x_2^L(t)], \end{aligned} \tag{2.24}$$

and similarly for $x_2$. Interestingly, it is possible to have $[y^L(t), y^U(t)] \subset [x_1^L(t) + x_2^L(t), x_1^U(t) + x_2^U(t)]$ and still fail to improve the bounds on $x_1$ or $x_2$ through these rearrangements. More generally, it can happen that the bounds on $x_1$ and $x_2$ are improved, but they still sum to an interval larger than $[y^L(t), y^U(t)]$. Thus, with this definition of $\mathcal{I}_G$, Method 4 is superior to Method 3. Modifying $\mathcal{I}_G$ so that Methods 3 and 4 are equivalent is straightforward for this example. However, in general there may be many sub-expressions involving multiple states that would benefit from the special treatment needed for $x_1 + x_2$ here, and addressing them automatically would greatly complicate $\mathcal{I}_G$. □

Developing an automated method for choosing effective $\mathbf{g}$ functions is a significant undertaking and is left as future research. Rather, in this chapter we demonstrate the manual construction of $\mathbf{g}$ for several case studies of practical interest. In doing so, we aim to provide insights towards a general method, and to demonstrate that extremely effective choices very often exist. Nevertheless, the preceding discussion suggests some broad approaches that are worth mentioning here. The first is simply to choose each $g_j$ in such a way that right-hand side function of the corresponding $y_j$ (i.e., $h_j \equiv \frac{\partial g_j}{\partial \mathbf{x}}\mathbf{f}$) admits nice algebraic simplifications. In particular, it is beneficial if this function can be written in a way that suffers minimally from the dependency problem, both in the conventional sense discusses in §2.3.2, and in the historical sense discussed in §2.3.4. The second strategy is to define $g_j(\mathbf{x})$ as a sub-expression appearing in the original ODEs that is either important to bound accurately, or problematic

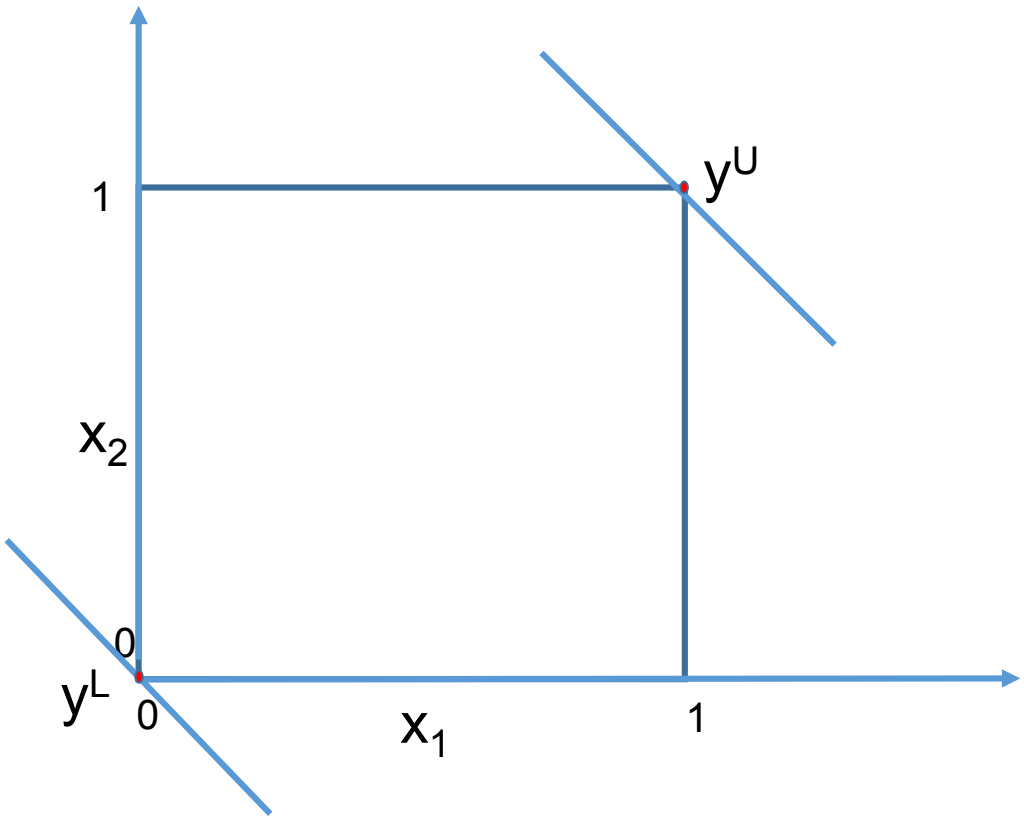


Figure 2.3: State bounds for $x_1$ and $x_2$ in Example 1 at $t_0$ (box), along with initial bounds for $y$ in Method 2. The set $\mathcal{I}_G(\mathcal{B}_4^L(Z(t_0)))$ is the singleton at the intersection of the box with the line $x_1 + x_2 = y^L$.

in the sense that it causes dependency in some $f_i$. With $y_j = g_j(\mathbf{x})$ defined, this sub-expression can be eliminated from $f_i$ by substituting $y_j$ in its place, leading to potential benefits of the type illustrated for Method 4 above.

### 2.4.1 Interpretation as a Non-Interval Enclosure Method

Let $X(t) = [\mathbf{x}^L(t), \mathbf{x}^U(t)]$ and $Y(t) = [\mathbf{y}^L(t), \mathbf{y}^U(t)]$ be state bounds for (2.12) computed using DI with manufactured model redundancy as described in Theorem 4. Although this bounding procedure uses only interval computations, it can be interpreted as propagating non-interval sets in the space of the original state variables. Namely,

$$\mathcal{X}(t) = \{\mathbf{z} \in X(t) : \mathbf{g}(\mathbf{z}) \in Y(t)\}. \tag{2.25}$$

Clearly, this enclosure need not be an interval and can be made arbitrarily complex by the choice of $\mathbf{g}$. From this point of view, DI with manufactured redundancy can be more readily compared with other state-of-the-art bounding approaches that make use of non-interval enclosures to combat conservatism.

First, we note that the ability to propagate nonconvex enclosures has so far been unique to Taylor model methods, and is thought to be an important factor in their ability to combat the wrapping effect for nonlinear systems. However, this requires second or higher order Taylor models, which require considerably more computational effort than intervals. In this regard, (2.25) provides an interesting alternative since it clearly provides nonconvex enclosures for appropriate choices of $\mathbf{g}$, and does so using only fast interval computations, albeit in a higher-dimensional state space. Moreover, the enclosures provided by Taylor model methods are always ranges of multivariate polynomials of fixed order, plus a remainder bound. In contrast, since $\mathbf{g}$ is

customizable and can be chosen based on the form of $\mathbf{f}$, (2.25) may also provide more flexible enclosures. Of course, these advantages are highly dependent on the choice of $\mathbf{g}$, and we cannot at present provide a general purpose method for this choice.

Second, we note that if we restrict ourselves to linear combinations of the original states, i.e., $\mathbf{g}(\mathbf{z}) = \mathbf{a}^{\mathrm{T}}\mathbf{z}$, then (2.25) is a polytope. In this case, DI with manufactured redundancy is very closely related to the polytopic DI method proposed in [17]. In that method, a number of vectors $\mathbf{a}_j^{\mathrm{T}}$ are chosen and DI is used to compute time-varying bounds $b_j(t)$ such that $\mathbf{a}_j^{\mathrm{T}}\mathbf{x}(t,\mathbf{p}) \leq b_j(t)$, $\forall (t,\mathbf{p}) \in I \times P$. Evidently, this is very similar to defining $y_j(t,\mathbf{p}) \equiv \mathbf{a}_j^{\mathrm{T}}\mathbf{x}(t,\mathbf{p})$ and bounding the augmented system to obtain the inequalities

$$y_j^L(t) \leq \mathbf{a}_j^{\mathrm{T}}\mathbf{x}(t,\mathbf{p}) \leq y_j^U(t), \quad \forall (t,\mathbf{p}) \in I \times P. \tag{2.26}$$

However, a significant difference is that the approach proposed here uses only fast interval computations. In contrast, the method in [17] requires solving an auxiliary system whose right-hand sides are defined through the solutions of linear programs. This is likely to be more accurate because the polytopic enclosure available at each $t$ can be enforced exactly, rather than approximately using an interval refinement operation $\mathcal{I}_G$. On the other hand, the linear programming approach is significantly more computationally demanding, and is inherently limited to polytopic enclosures. In contrast, DI with manufactured redundancy directly extends to nonlinear $\mathbf{g}$, and hence nonconvex enclosures, while retaining high efficiency.

Nevertheless, in the numerical experiments in §2.6, we only consider linear $\mathbf{g}$ functions for all but one example. This is because, although Theorem 4 permits nonlinear $\mathbf{g}$, it requires a valid definition of $\mathcal{I}_G$ for such $\mathbf{g}$ satisfying Assumption 2. We propose one such $\mathcal{I}_G$ for a specific case in §2.6, but we leave the general case for

future work.

In the next section, we present a more efficient version of the interval refinement operation $\mathcal{I}_G$ of Algorithm 1, and develop new preconditioning strategies that enable this $\mathcal{I}_G$ to make much more effective use of linear $\mathbf{g}$. Numerical results in §2.6 suggest that these strategies can make $\mathcal{I}_G$ nearly as effective as the linear programming approach in [17] at significantly lower cost.

## 2.5 Improved Methods for ODEs with Affine Solution Invariants

This sections considers new strategies for further increasing the efficiency and reducing the conservatism of the bounding system (2.10) applied to ODEs (4.1a) satisfying

$$\mathbf{x}(t,\mathbf{p}) \in G \equiv \{\mathbf{z} : \mathbf{M}\mathbf{z} = \mathbf{b}\}, \quad \forall (t,\mathbf{p}) \in I \times P, \tag{2.27}$$

for some known $\mathbf{M} \in \mathbb{R}^{n_m \times n_x}$ and $\mathbf{b} \in \mathbb{R}^{n_m}$. Of course, in light of §2.4, it is not necessary to assume that such an enclosure is available for the original ODEs of interest. In general, any ODEs (4.1a) can be embedded in an augmented system (2.12) satisfying (2.27) by including the additional states $\mathbf{y} = \mathbf{A}\mathbf{x}$ and defining $\mathbf{M} = [-\mathbf{A}\ \mathbf{I}]$ and $\mathbf{b} = \mathbf{0}$. For simplicity, we denote all state variables by $\mathbf{x}$ in this section, regardless of whether or not they are states of the original system. Given that $G$ satisfying (2.27) is available, our interest is to determine how to optimally exploit it in a bounding procedure like (2.10). We provide only a heuristic answer here, but one that results in marked improvements. In addition to providing improved bounds for fixed $G$, this also provides important insights for manufacturing $G$ (via the choice of $\mathbf{A}$ above),

and provides a firm basis for comparing alternative choices.

### 2.5.1 Preconditioning Heuristics

Consider the interval refinement operation $\mathcal{I}_G$ defined by Algorithm 1. By Assumption 2, this operation satisfies

$$\mathcal{I}_G(Z) \supset \{\mathbf{z} \in Z : \mathbf{M}\mathbf{z} = \mathbf{b}\}, \quad \forall Z \in \mathbb{IR}^{n_x}. \tag{2.28}$$

The purpose of this subsection is to demonstrate that preconditioning the constraint system $\mathbf{M}\mathbf{z} = \mathbf{b}$ can have a profound impact on the accuracy of this enclosure, and hence on the accuracy of the state bounds computed via (2.10), and to present a new preconditioning strategy to addresses this issue. The central issue necessitating preconditioning is illustrated in Example 2, while the affect on state bounds is demonstrated in Example 3.

**Example 2** *Consider the following two sets of linear constraints:*

$$\underbrace{\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}}_{\mathbf{M}} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 \\ 0 \end{bmatrix}}_{\mathbf{b}} \qquad \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{\mathbf{F}} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 \\ 0 \end{bmatrix}}_{\mathbf{d}} \tag{2.29}$$

*Clearly, these constraints are equivalent and have the origin as their unique solution. Now, consider applying $\mathcal{I}_G(Z)$ to refine the interval $Z = [-1, 1] \times [-1, 1]$ as in Algorithm 1. As shown in Figure 2.4, this interval cannot be refined by considering rearrangements of the constraints $\mathbf{M}\mathbf{z} = \mathbf{b}$, as long as the constraints are considered one at a time, as they are in Algorithm 1. In contrast Algorithm 1 readily refines $Z$ to the singleton $\{\mathbf{0}\}$ using $\mathbf{F}\mathbf{z} = \mathbf{d}$.* □

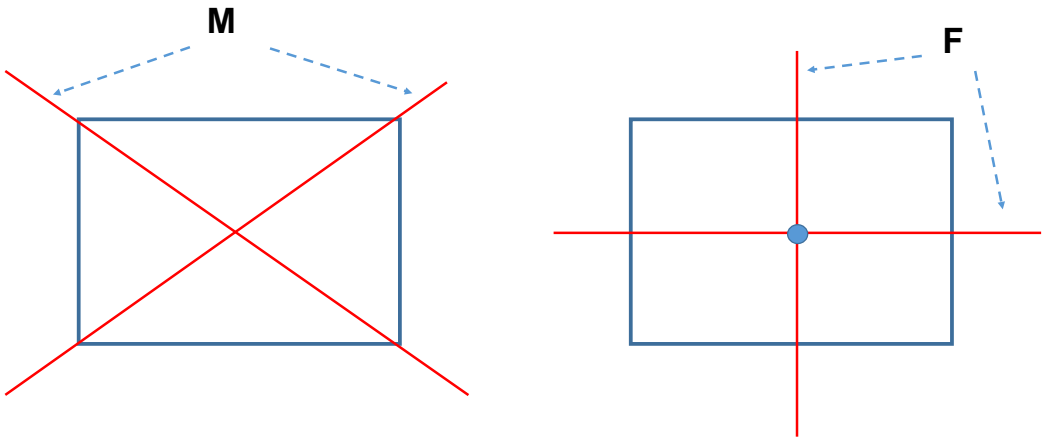


Figure 2.4: The interval $Z$ in Example 2 (rectangle) with lines corresponding to the equations $\mathbf{Mz} = \mathbf{b}$ (left) and $\mathbf{Fz} = \mathbf{d}$ (right).

**Example 3** *Consider the enzymatic reaction network described by [64]:*

$$\mathrm{A} + \mathrm{F} \rightleftharpoons \mathrm{F} : \mathrm{A} \to \mathrm{F} + \mathrm{A}',$$

$$\mathrm{A}' + \mathrm{R} \rightleftharpoons \mathrm{R} : \mathrm{A}' \to \mathrm{R} + \mathrm{A}.$$

*The dynamics can be modeled by the following ODEs:*

$$\begin{aligned}
\dot{x}_{\mathrm{A}} &= -k_1 x_{\mathrm{A}} x_{\mathrm{F}} + k_2 x_{\mathrm{F:A}} + k_6 x_{\mathrm{R:A'}} \\
\dot{x}_{\mathrm{F}} &= -k_1 x_{\mathrm{A}} x_{\mathrm{F}} + k_2 x_{\mathrm{F:A}} + k_3 x_{\mathrm{F:A}} \\
\dot{x}_{\mathrm{F:A}} &= k_1 x_{\mathrm{A}} x_{\mathrm{F}} - k_2 x_{\mathrm{F:A}} - k_3 x_{\mathrm{F:A}} \\
\dot{x}_{\mathrm{A'}} &= k_3 x_{\mathrm{F:A}} - k_4 x_{\mathrm{A'}} x_{\mathrm{R}} + k_5 x_{\mathrm{R:A'}} \\
\dot{x}_{\mathrm{R}} &= -k_4 x_{\mathrm{A'}} x_{\mathrm{R}} + k_5 x_{\mathrm{R:A'}} + k_6 x_{\mathrm{R:A'}} \\
\dot{x}_{\mathrm{R:A'}} &= k_4 x_{\mathrm{A'}} x_{\mathrm{R}} - k_5 x_{\mathrm{R:A'}} - k_6 x_{\mathrm{R:A'}}
\end{aligned} \tag{2.30}$$

*Let* $I = [0, 0.04]$ *(s),* $\mathbf{x}_0 = (34, 20, 0, 0, 16, 0)$ *(M), and let the uncertain rate parameters* $\mathbf{k} = (k_1, \ldots, k_6)$ *lie in the set* $P = [\hat{\mathbf{k}}, 10\hat{\mathbf{k}}]$ *with* $\hat{\mathbf{k}} = (0.1, 0.033, 16, 5, 0.5, 0.3)$.

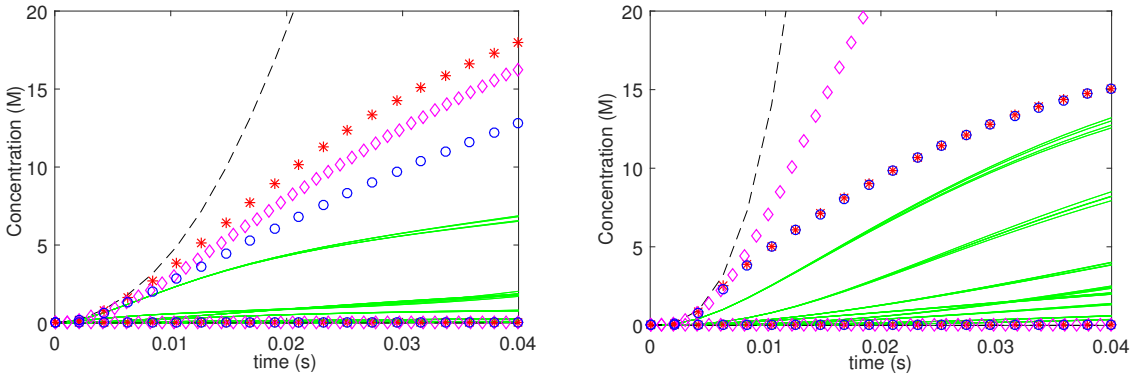


Figure 2.5: State bounds for $x_{\mathrm{A}'}$ (left) and $x_{\mathrm{RA}'}$ (right) from Example 3, computed using standard DI (dashed), and using (2.10) and Algorithm 1 with $\mathbf{M}$ (red star) and $\mathbf{F}$ (magenta diamond) in (2.31). Blue circles are obtained using the new preconditioning method described in §2.5.1. Solid lines are real solutions.

*Chemical reaction systems of this type are well-known to satisfy the affine solution invariants* $\mathbf{Mx}(t,\mathbf{p}) = \mathbf{Mx}_0$ *for any* $\mathbf{M}$ *whose rows lie in the left null space of the stoichiometry matrix* $\mathbf{S}$ *[12]. Of course, this choice is not unique. Figure 2.5 shows state bounds for* (2.30) *computed via* (2.10) *using* $\mathcal{I}_G$ *as in Algorithm 1 with two different choices of* $\mathbf{M}$*. The first has rows that form an orthonormal basis for the left null space of* $\mathbf{S}$ *obtained using the MATLAB subroutine* `null`*. The second, denoted* $\mathbf{F}$*, is physically motivated by conservation laws:*

$$\mathbf{M} = \begin{bmatrix} -0.48 & -0.14 & -0.62 & -0.48 & 0.24 & -0.24 \\ -0.31 & 0.75 & 0.43 & -0.31 & 0.15 & -0.15 \\ 0 & 0 & 0 & 0 & 0.70 & 0.70 \end{bmatrix} \tag{2.31}$$

$$\mathbf{F} = \begin{bmatrix} 0 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & -1 \\ 1 & -1 & 0 & 1 & -1 & 0 \end{bmatrix}$$

*Figure 2.5 clearly demonstrates that this choice has a significant impact on the accuracy of the bounds.* □

In light of the previous examples, we would like to develop an algorithm for preconditioning a given set of constraints $\mathbf{Mz} = \mathbf{b}$ satisfying (2.27), before solving the bounding system (2.10), to obtain another set $\mathbf{Fz} = \mathbf{d}$ that enables $\mathcal{I}_G$ to refine

intervals $Z \in \mathbb{R}^{n_x}$ more effectively. To begin, consider using $\mathbf{Mz} = \mathbf{b}$ to refine a single element $Z_j$. Since $\mathcal{I}_G$ considers rearrangements of only one equation at a time, our basic strategy is to provide $\mathcal{I}_G$ with one or more equations of the form $\boldsymbol{\mu}^{\mathrm{T}}\mathbf{Mz} = \boldsymbol{\mu}^{\mathrm{T}}\mathbf{b}$ with $\boldsymbol{\mu} \in \mathbb{R}^{n_m}$ that are specially designed to be effective for refining $Z_j$ (each of these will be a row of $\mathbf{Fz} = \mathbf{d}$). Of course, we aim to provide such equations for all choices of $j$, which raises an interesting observation: $\mathbf{F}$ may have more rows than $\mathbf{M}$. Clearly, the rows of such an $\mathbf{F}$ will be linearly dependent. Thus, this is yet another use of redundancy to improve the accuracy of interval operations.

Denoting the $j^{\text{th}}$ column of $\mathbf{M}$ by $\mathbf{m}_j$ and choosing any $\boldsymbol{\mu} \in \mathbb{R}^{n_m}$ with $\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j \neq 0$, rearranging $\boldsymbol{\mu}^{\mathrm{T}}\mathbf{Mz} = \boldsymbol{\mu}^{\mathrm{T}}\mathbf{b}$ gives

$$z_j = \frac{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{b}}{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j} - \sum_{k \neq j} \frac{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k}{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j} z_k. \tag{2.32}$$

Thus, potentially improved bounds for $z_j$ are given by

$$\hat{z}_j^L = \frac{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{b}}{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j} - \sum_{k \neq j} \max\left(\frac{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k}{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j} z_k^L, \frac{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k}{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j} z_k^U\right), \tag{2.33}$$

$$\hat{z}_j^U = \frac{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{b}}{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j} - \sum_{k \neq j} \min\left(\frac{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k}{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j} z_k^L, \frac{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k}{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j} z_k^U\right). \tag{2.34}$$

Now consider maximizing and minimizing (2.33) and (2.34), respectively, over the set $\{\boldsymbol{\mu} : \boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j \neq 0\}$ to obtain the solutions $\boldsymbol{\mu}_{z_j^L}$ and $\boldsymbol{\mu}_{z_j^U}$. These vectors encode combinations of the constraints $\mathbf{Mz} = \mathbf{b}$ that are optimal for improving the $j^{\text{th}}$ lower and upper bounds of the given interval $Z$ via (2.33) and (2.34), respectively. Unfortunately, these solutions depend on $Z_{k \neq j}$, which is problematic because $\mathcal{I}_G$ will be evaluated with many different interval arguments during the solution of (2.10). Moreover, these arguments are not known prior to solving (2.10). Thus, in what

follows, we develop a method for computing choices of $\boldsymbol{\mu}$ that are provably optimal for entire classes of intervals $Z$. Although there is no guarantee that the arguments of $\mathcal{I}_G$ in (2.10) will lie in these classes, we expect that these procedures will produce more robust choices of $\boldsymbol{\mu}$.

The classes of intervals we consider are defined as follows. These classes are parameterized by $\mathbf{r}$ and contain all intervals with centers in $G$ and radii proportional to $\mathbf{r}$.

**Definition 7** *For any* $\mathbf{M} \in \mathbb{R}^{n_m \times n_x}$, $\mathbf{b} \in \mathbb{R}^{n_m}$, *and* $\mathbf{r} \in \mathbb{R}^{n_x}$, *define the following subset of* $\mathbb{IR}^{n_x}$, *where* $\mathbf{x}_m$ *and* $\mathbf{x}_r$ *denote the midpoint and radius of* $X$, *respectively:*

$$\mathcal{X}(\mathbf{M}, \mathbf{b}, \mathbf{r}) \equiv \left\{ X \in \mathbb{IR}^{n_x} : \begin{array}{c} \mathbf{M}\mathbf{x}_m = \mathbf{b}, \\ \exists \alpha \in \mathbb{R}_+ : \mathbf{x}_r = \alpha \mathbf{r} \end{array} \right\}. \tag{2.35}$$

The following theorem describes optimal choices of $\boldsymbol{\mu}$ for refining the $j^{\text{th}}$ component of an interval $Z \in \mathbb{IR}^{n_x}$ using $\mathcal{I}_G$. First, note that although the specific arguments of $\mathcal{I}_G$ in (2.10) cannot be known *a priori*, there is one distinctive feature of all such arguments that proves useful - they are all of the form $\mathcal{B}_i^{L/U}(X)$, and hence are degenerate in one dimension. Not surprisingly, the effectiveness of a given $\boldsymbol{\mu}$ for refining the $j^{\text{th}}$ component of $\mathcal{B}_i^L(X)$ can depend strongly on $i$, and can be different for $\mathcal{B}_i^U(X)$. Thus, it proves useful to design $\boldsymbol{\mu}$'s corresponding to each $\mathcal{B}_i^{L/U}(X)$. The following theorem concerns $\mathcal{B}_i^L(X)$, while Theorem 6 concerns $\mathcal{B}_i^U(X)$.

**Theorem 5** *Choose any* $\mathbf{r} \in \mathbb{R}^{n_x}$ *and* $i, j \in \{1, \ldots, n_x\}$, $i \neq j$. *Assuming that* (2.36)

*and* (2.37) *are bounded, let*

$$\boldsymbol{\mu}_{z_j^L} \in \underset{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j=1}{\arg\max} \left( \boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_i r_i - \textstyle\sum_{k\notin\{j,i\}} |\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k r_k| \right), \tag{2.36}$$

$$\boldsymbol{\mu}_{z_j^U} \in \underset{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j=1}{\arg\min} \left( \boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_i r_i + \textstyle\sum_{k\notin\{j,i\}} |\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k r_k| \right), \tag{2.37}$$

$$\boldsymbol{\mu}_{z_j^W} \in \underset{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j=1}{\arg\min} \left( \textstyle\sum_{k\notin\{j,i\}} |\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k r_k| \right). \tag{2.38}$$

*Let* $U = \{\boldsymbol{\mu} : \boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j \neq 0\}$. *Then, for any* $Z \equiv \mathcal{B}_i^L(X)$ *with* $X \in \mathcal{X}(\mathbf{M}, \mathbf{b}, \mathbf{r})$*:*

1. $\boldsymbol{\mu}_{z_j^L}$ *maximizes the right-hand side of* (2.33) *on* $U$*,*
2. $\boldsymbol{\mu}_{z_j^U}$ *minimizes the right-hand side of* (2.34) *on* $U$*, and*
3. $\boldsymbol{\mu}_{z_j^W}$ *minimizes the width* $\hat{z}_j^U - \hat{z}_j^L$ *in* (2.33)–(2.34) *on* $U$*.*

**Proof** To prove Claim 1, we first show that (2.36) is related to the dual of the following linear program:

$$\min_{\mathbf{z}} \{ z_j : \mathbf{M}\mathbf{z} = \mathbf{b},\ z_k^L \leq z_k \leq z_k^U,\ \forall k \neq j \}. \tag{2.39}$$

Applying LP duality, the dual of (2.39) is:

$$\max_{\boldsymbol{\mu},\eta_k,\gamma_k} \quad \boldsymbol{\mu}^{\mathrm{T}}\mathbf{b} + \sum_{k\neq j} (\gamma_k z_k^L - \eta_k z_k^U) \tag{2.40}$$

$$\text{s.t.} \quad \boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j = 1 \tag{2.41}$$

$$\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k = (\eta_k - \gamma_k),\ \forall k \neq j \tag{2.42}$$

$$\gamma_k, \eta_k \geq 0,\ \forall k \neq j \tag{2.43}$$

For any fixed $\boldsymbol{\mu}$, maximizing the term $(\gamma_k z_k^L - \eta_k z_k^U)$ with respect to $\eta_k$ and $\gamma_k$ subject

to (2.42)–(2.43) gives $-\max(\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k z_k^L, \boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k z_k^U)$. Thus, (2.40) is equivalent to

$$\max_{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j=1} \quad \boldsymbol{\mu}^{\mathrm{T}}\mathbf{b} - \sum_{k\neq j} \max(\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k z_k^L, \boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k z_k^U). \tag{2.44}$$

Alternatively, using the midpoint and radius of $Z$ to write $z_k^L = z_{m,k} - z_{r,k}$ and $z_k^U = z_{m,k} + z_{r,k}$ gives

$$\max_{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j=1} \quad \boldsymbol{\mu}^{\mathrm{T}}\mathbf{b} - \sum_{k\neq j} \boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k z_{m,k} - \sum_{k\neq j} |\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k| z_{r,k}. \tag{2.45}$$

Adding and subtracting $\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j z_{m,j}$ gives

$$\max_{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j=1} \quad \boldsymbol{\mu}^{\mathrm{T}}(\mathbf{b} - \mathbf{M}\mathbf{z}_m) + z_{m,j} - \sum_{k\neq j} |\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k| z_{r,k}. \tag{2.46}$$

Now, since $Z = \mathcal{B}_i^L(X)$, it can be derived that $\mathbf{z}_m = \mathbf{x}_m - \mathbf{e}_i x_{r,i}$ and $\mathbf{z}_r = \mathbf{x}_r - \mathbf{e}_i x_{r,i}$, where $\mathbf{e}_i$ is the $i^{\text{th}}$ unit vector. Substituting these into (2.46) gives

$$\begin{aligned} \max_{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j=1} \quad & \boldsymbol{\mu}^{\mathrm{T}}(\mathbf{b} - \mathbf{M}\mathbf{x}_m) + \boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_i x_{r,i} + x_{m,j} \\ & - \sum_{k\notin\{i,j\}} |\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k x_{r,k}|. \end{aligned} \tag{2.47}$$

Also, with the assumption $X \in \mathcal{X}(\mathbf{M}, \mathbf{b}, \mathbf{r})$, we have $\mathbf{b} = \mathbf{M}\mathbf{x}_m$ and $\mathbf{x}_r = \alpha\mathbf{r}$, thus (2.47) becomes

$$\max_{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j=1} \quad \boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_i \alpha r_i + x_{m,j} - \alpha \sum_{k\notin\{i,j\}} |\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k r_k|. \tag{2.48}$$

Finally, since $x_{m,j}$ and $\alpha > 0$ are constants, $\boldsymbol{\mu}_{z_j^L}$ satisfies (2.36) if and only if it is a solution of (2.48).

Now, to prove Claim 1, let $\boldsymbol{\mu}_{z_j^L}$ satisfy (2.36) and let $Q(\boldsymbol{\mu})$ denote the right-hand side of (2.33). Since (2.36) is bounded, (2.48) must also be bounded, and by duality (2.39) must be feasible. Let $\mathbf{z}^*$ be a solution of (2.39). Next, note that (2.32) holds for any $\mathbf{z}$ with $\mathbf{M}\mathbf{z} = \mathbf{b}$ and any $\boldsymbol{\mu} \in U$, and therefore $\max_{\boldsymbol{\mu}\in U} Q(\boldsymbol{\mu}) \leq z_j^*$. But $\boldsymbol{\mu}_{z_j^L}^{\mathrm{T}}\mathbf{m}_j = 1$ implies that $\boldsymbol{\mu}_{z_j^L}^{\mathrm{T}} \in U$, and so it suffices to show that $z_j^* = Q(\boldsymbol{\mu}_{z_j^L}^{\mathrm{T}})$. As noted above, $\boldsymbol{\mu}_{z_j^L}$ must be a maximizer of (2.48), which is equivalent to (2.44) by the preceding derivations. Thus, by strong duality,

$$z_j^* = \boldsymbol{\mu}_{z_j^L}^{\mathrm{T}}\mathbf{b} - \sum_{k\neq j} \max(\boldsymbol{\mu}_{z_j^L}^{\mathrm{T}}\mathbf{m}_k z_k^L, \boldsymbol{\mu}_{z_j^L}^{\mathrm{T}}\mathbf{m}_k z_k^U) \tag{2.49}$$

$$= Q(\boldsymbol{\mu}_{z_j^L}). \tag{2.50}$$

This establishes Claim 1. The remaining two claims are proven analogously and are omitted for brevity. □

**Theorem 6** *Choose any* $\mathbf{r} \in \mathbb{R}^{n_x}$ *and* $i, j \in \{1, \ldots, n_x\}$, $i \neq j$. *Assuming that* (2.51) *and* (2.52) *are bounded, let*

$$\boldsymbol{\mu}_{z_j^L} \in \underset{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j=1}{\arg\max} \left(-\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_i r_i - \textstyle\sum_{k\notin\{j,i\}} |\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k r_k|\right), \tag{2.51}$$

$$\boldsymbol{\mu}_{z_j^U} \in \underset{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j=1}{\arg\min} \left(-\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_i r_i + \textstyle\sum_{k\notin\{j,i\}} |\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k r_k|\right), \tag{2.52}$$

$$\boldsymbol{\mu}_{z_j^W} \in \underset{\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j=1}{\arg\min} \left(\textstyle\sum_{k\notin\{j,i\}} |\boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_k r_k|\right). \tag{2.53}$$

*Let* $U = \{\boldsymbol{\mu} : \boldsymbol{\mu}^{\mathrm{T}}\mathbf{m}_j \neq 0\}$. *Then, for any* $Z \equiv \mathcal{B}_i^U(X)$ *with* $X \in \mathcal{X}(\mathbf{M}, \mathbf{b}, \mathbf{r})$*:*

1. $\boldsymbol{\mu}_{z_j^L}$ *maximizes the right-hand side of* (2.33) *on* $U$*,*
2. $\boldsymbol{\mu}_{z_j^U}$ *minimizes the right-hand side of* (2.34) *on* $U$*, and*
3. $\boldsymbol{\mu}_{z_j^W}$ *minimizes the width* $\hat{z}_j^U - \hat{z}_j^L$ *in* (2.33)–(2.34) *on* $U$*.*

**Proof** The proof is analogous to that of Theorem 5. □

Based on the preceding two theorems, we use the following preconditioning method. First, we choose $\mathbf{r} \in \mathbb{R}^{n_x}$, which is a rough estimate of the relative radius of the intervals $X(t) = [\mathbf{x}^L(t), \mathbf{x}^U(t)]$ that will be generated by (2.10). If no physical information is available to guide this choice, then we specify $\mathbf{r} = \mathbf{1}$. Another option is to simulate a single trajectory and choose $\mathbf{r}$ based on the relative magnitudes of the states. Note that the choice of $\mathbf{r}$ determines the class of intervals for which the $\boldsymbol{\mu}$'s in Theorems 5–6 are optimal, but any choice will lead to valid state bounds (because $\boldsymbol{\mu}^{\mathrm{T}}\mathbf{M}\mathbf{z} = \boldsymbol{\mu}^{\mathrm{T}}\mathbf{b}$ is a valid constraint for any $\boldsymbol{\mu}$).

Given $\mathbf{r}$, we generate two preconditioned forms of the constraints $\mathbf{M}\mathbf{z} = \mathbf{b}$ for each $i \in \{1, \ldots, n_x\}$. The first corresponds to $\mathcal{B}_i^L$ and is denoted $\mathbf{F}_i^L\mathbf{z} = \mathbf{d}_i^L$. To form it, we solve the LPs (2.36)–(2.38) for every $j \neq i$. Then, we set

$$\mathbf{F}_i^L = \begin{bmatrix} \boldsymbol{\mu}_{z_1^L}^{\mathrm{T}} \\ \boldsymbol{\mu}_{z_1^U}^{\mathrm{T}} \\ \boldsymbol{\mu}_{z_1^W}^{\mathrm{T}} \\ \vdots \\ \boldsymbol{\mu}_{z_{n_x}^L}^{\mathrm{T}} \\ \boldsymbol{\mu}_{z_{n_x}^U}^{\mathrm{T}} \\ \boldsymbol{\mu}_{z_{n_x}^W}^{\mathrm{T}} \end{bmatrix} \mathbf{M} \quad \text{and} \quad \mathbf{d}_i^L = \begin{bmatrix} \boldsymbol{\mu}_{z_1^L}^{\mathrm{T}} \\ \boldsymbol{\mu}_{z_1^U}^{\mathrm{T}} \\ \boldsymbol{\mu}_{z_1^W}^{\mathrm{T}} \\ \vdots \\ \boldsymbol{\mu}_{z_{n_x}^L}^{\mathrm{T}} \\ \boldsymbol{\mu}_{z_{n_x}^U}^{\mathrm{T}} \\ \boldsymbol{\mu}_{z_{n_x}^W}^{\mathrm{T}} \end{bmatrix} \mathbf{b}. \tag{2.54}$$

The second corresponds to $\mathcal{B}_i^U$, is denoted $\mathbf{F}_i^U\mathbf{z} = \mathbf{d}_i^U$, and is formed analogously from the solutions of LPs (2.51)–(2.53). This process is then repeated for each $i$. During the solution of (2.10), whenever $\mathcal{I}_G$ is evaluated on an interval of the form $\mathcal{B}_i^L(X(t))$, the corresponding constraint system $\mathbf{F}_i^L\mathbf{z} = \mathbf{d}_i^L$ is used in place of $\mathbf{M}\mathbf{z} = \mathbf{b}$, and similarly for $\mathcal{B}_i^U(X(t))$.

Overall, this procedure requires solving $3n_x(n_x - 1)$ LPs. However, this is done only once, prior to the solution of (2.10). In contrast, the method proposed in

[17] requires solving $4n_x$ LPs at every time step during numerical integration of the bounding system. Moreover, in the context of B&B global optimization, these LPs would only need to be solved in the root node, and the solutions could then be used repeatedly during the solution of (2.10) in every other node of the B&B tree.

Figure 2.5 shows the bounds for Example 3 using the preconditioning strategy outlined above. Clearly, this procedure leads to improved bounds, particularly when compared to the original $\mathbf{M}$ matrix (recall that $\mathbf{F}$ in (2.31) was known from physical insights and is not available in general).

Note that, if $\mathcal{I}_G(Z)$ as in Algorithm 1 is applied to the constraints $\mathbf{F}_i^L\mathbf{z} = \mathbf{d}_i^L$, then e.g. the first constraint $\boldsymbol{\mu}_{z_j^L}^{\mathrm{T}}\mathbf{M}\mathbf{z} = \boldsymbol{\mu}_{z_j^L}^{\mathrm{T}}\mathbf{b}$ will be used to attempt to refine every $Z_k$ that has a nonzero coefficient in the constraint, even though this constraint was designed only to refine $z_j^L$. Thus, it seems that many computations can be avoided in $\mathcal{I}_G(Z)$ by only using each constraint for its intended purpose, likely with little performance degradation. However, as shown in the next section, $\mathcal{I}_G(Z)$ can be implemented more efficiently than in Algorithm 1, and in this new implementation the cost of refining $z_j^L$ is only marginally smaller than that of refining all possible bounds using $\boldsymbol{\mu}_{z_j^L}^{\mathrm{T}}\mathbf{M}\mathbf{z} = \boldsymbol{\mu}_{z_j^L}^{\mathrm{T}}\mathbf{b}$. At the same time, these additional refinements can prove beneficial, particularly when the bounding system (2.10) produces intervals $X(t)$ for which the designed preconditioners are not optimal.

### 2.5.2 A More Efficient Implementation of $\mathcal{I}_G$

In this section, a more efficient version of Algorithm 1 from [64] is developed. First, note that the complexity of Algorithm 1 is $O(n_m n_x^2)$. Here, we reduce this to $O(n_m n_x)$ (with a prefactor about $4\times$ as large) by eliminating some repeated computations. Specifically, for a fixed equation $\mathbf{m}_i^{\mathrm{T}}\mathbf{z} = \mathbf{b}_i$, Algorithm 1 computes the

following sums independently for every $j$:

$$\zeta \leftarrow \frac{b_i}{m_{ij}} + \sum_{k=1,k\neq j}^{n_x} \min(-\frac{m_{ik}}{m_{ij}} z_k^L, -\frac{m_{ik}}{m_{ij}} z_k^U), \tag{2.55}$$

$$\gamma \leftarrow \frac{b_i}{m_{ij}} + \sum_{k=1,k\neq j}^{n_x} \max(-\frac{m_{ik}}{m_{ij}} z_k^L, -\frac{m_{ik}}{m_{ij}} z_k^U), \tag{2.56}$$

Clearly, for any two choices of $j$, these sums have $n_x - 2$ terms in common (up to a scalar multiple). These repeated computations can be avoided by instead computing $\alpha^L$ and $\alpha^U$ as defined in lines 4–5 of Algorithm 2. The complexity of computing $\alpha^L$ and $\alpha^U$ is nearly the same as that of computing (2.55)–(2.56) once. However, once $\alpha^L$ and $\alpha^U$ are known, $\zeta$ and $\gamma$ can be computed easily for any $j$ using lines 10–11 in Algorithm 2. Moreover, after $z_j^L$ and $z_j^U$ are updated using $\zeta$ and $\gamma$, $\alpha^L$ and $\alpha^U$ can be easily updated as shown in lines 14–15.

With these changes, the complexity of Algorithm 1 is reduced by a factor of $n_x$. Moreover, Algorithm 2 would produce exactly the same results as Algorithm 1, but for one caveat. In order to avoid repeated sums as described above, Algorithm 2 loops through the elements of $\mathbf{M}$ one row at a time, rather than one column at a time as in Algorithm 1. This means that Algorithm 2 will consider rearranging the constraint in the first row of $\mathbf{Mz} = \mathbf{b}$ for $z_1$, then $z_2$, and so on until $z_{n_x}$, before moving on to the second row. In contrast, Algorithm 1 considers rearranging all constraints for $z_1$ before moving on to $z_2$. In general, the results will be different, but there is no reason to believe that one ordering is better than the other.

**Algorithm 2** A faster version of $\mathcal{I}_G$ from [64]

**function** FastIG($\mathbf{z}^L, \mathbf{z}^U, X_{\text{nat}}, \mathbf{M}, \mathbf{b}$, tol)
  $[\mathbf{z}^L, \mathbf{z}^U] \leftarrow [\mathbf{z}^L, \mathbf{z}^U] \cap X_{\text{nat}}$
  **for** $i \leftarrow 1, n_m$ **do**
    $\alpha^L \leftarrow b_i - \sum_{k=1}^{n_x} \max(m_{ik} z_k^L, m_{ik} z_k^U)$
    $\alpha^U \leftarrow b_i - \sum_{k=1}^{n_x} \min(m_{ik} z_k^L, m_{ik} z_k^U)$
    **for** $j \leftarrow 1, n_x$ **do**
      **if** $|m_{ij}| >$ tol **then**
        $\alpha^L \leftarrow \alpha^L + \max(m_{ij} z_j^L, m_{ij} z_j^U)$
        $\alpha^U \leftarrow \alpha^U + \min(m_{ij} z_j^L, m_{ij} z_j^U)$
        $\zeta \leftarrow \min(\alpha^L / m_{ij}, \alpha^U / m_{ij})$
        $\gamma \leftarrow \max(\alpha^L / m_{ij}, \alpha^U / m_{ij})$
        $z_j^L \leftarrow \text{mid}(z_j^L, z_j^U, \zeta)$
        $z_j^U \leftarrow \text{mid}(z_j^L, z_j^U, \gamma)$
        $\alpha^L \leftarrow \alpha^L - \max(m_{ij} z_j^L, m_{ij} z_j^U)$
        $\alpha^U \leftarrow \alpha^U - \min(m_{ij} z_j^L, m_{ij} z_j^U)$
      **end if**
    **end for** ▷ $j \leftarrow 1, n_x$
  **end for** ▷ $i \leftarrow 1, n_u$
  **return** $[\mathbf{z}^L, \mathbf{z}^U]$
**end function**

## 2.6 Numerical examples

In this section, we present six examples to illustrate the advantages of introducing new redundant state variables as describe in §2.4. In each example, at least two methods are compared. The first is standard differential inequalities (SDI), which directly solves the bounding system (2.7), and does not use any solution invariants. If the system naturally obeys some pre-existing solution invariants, then bounds are also computed using only these invariants by solving (2.10) with $\mathcal{I}_G$ as defined in Algorithm 2. Finally, we consider the addition of redundant state variables and again apply (2.10) with Algorithm 2, using both the natural and manufactured invariants. We report wall clock times for all methods as implemented in `MATLAB` using the numerical integrator `CVODE` in the Sundials Toolbox with default settings [21]. All computations were implemented on a Dell Precision T3610 workstation with an Intel

Xeon E5-1607 v2 @ 3.00 GHz. For every problem with linear invariants $\mathbf{Mz} = \mathbf{b}$, the preconditioning method described in §2.5.1 was applied using `CPLEX` vesion 12.3 to solve all LPs. Unless otherwise stated, this preconditioning is done with $\mathbf{r} = \mathbf{1}$. The wall clock times for preconditioning are not included in the examples below, but ranged from 0.012 s for Example 8 to 0.45 s for Example 5. We stress that these times are relatively unimportant in our applications of interest because they are incurred only once, while the results can be used in repeated bounding computations. For example, in branch-and-bound global dynamic optimization, the cost of preconditioning is potentially shared between many thousands of bounding computations [62].

**Example 4** *Consider again the reaction network described in Example 3. In [64], state bounds for this problem have already been significantly improved using the three pre-existing solution invariants in this system. To make further improvements, one redundant state variable* $y = -x_{\mathrm{A}} + x_{\mathrm{F}}$ *was introduced. From* (2.30)*, it can be seen that the corresponding ODE enjoys two significant term cancellations, and can be expressed as* $\dot{y} = k_3 x_{\mathrm{F:A}} - k_6 x_{\mathrm{R:A'}}$*. The addition of* $y$ *also introduces one more solution invariant. Thus, with* $\mathbf{M}$ *as defined in* (2.31)*, the augmented system satisfies*

$$\begin{bmatrix} -1 & 1 & 0 & \multicolumn{3}{c}{\multirow{2}{*}{$\mathbf{M}$}} & 0 \\ 0 & 0 & 0 & & & & -1 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ y \end{bmatrix} = \begin{bmatrix} \mathbf{M x_0} \\ 0 \end{bmatrix}. \tag{2.57}$$

*All state variable are nonnegative and bounded above by* $\bar{\mathbf{x}} = (34, 20, 20, 34, 16, 16)$ *[64], and consequently* $y \in [-34, 20]$*. The relative radius estimate* $\mathbf{r}$ *required for preconditioning was chosen as the radius of these natural bounds,* $\mathbf{r} = (17, 10, 10, 17, 8, 8, 27)$.

*Figure 2.6 shows that the bounds computed using the augmented system are significantly sharper than those computed using only pre-existing solution invariants. Thus, the proposed technique is effective even for systems that already benefit from a significant number of invariants. The cost of a single trajectory was* 0.004 *s, while*

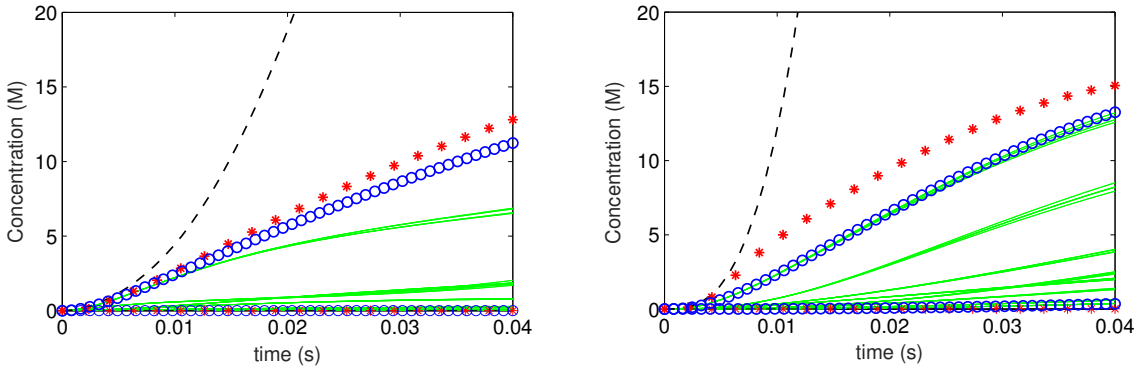


Figure 2.6: State bounds for $x_{\mathrm{A}'}$ (left) and $x_{\mathrm{RA}'}$ (right) from (2.30) computed using SDI (dashed), DI with pre-existing invariants (stars), and DI with both pre-existing and manufactured invariants (circles). Solid lines are real trajectories.

*computing bounds required* 0.014 *s for SDI,* 0.055 *s for DI with pre-existing model redundancy, and* 0.073 *s for DI with both existing and manufactured model redundancy.*

**Example 5** *In contrast to Example 4, this example shows the advantages of introducing model redundancy for a problem with few pre-existing invariants. Consider the following series reaction in batch reactor:*

$$\mathrm{A} \to \mathrm{B} \to \mathrm{C} \to \mathrm{D} \to \mathrm{E} \to \mathrm{F} \to \mathrm{G} \to \mathrm{H} \to \mathrm{I} \to \mathrm{J}.$$

*The corresponding system of ODEs is:*

$$\begin{aligned}
\dot{x}_{\mathrm{A}} &= -k_1 x_{\mathrm{A}} & \dot{x}_{\mathrm{F}} &= k_5 x_{\mathrm{E}} - k_6 x_{\mathrm{F}} \\
\dot{x}_{\mathrm{B}} &= k_1 x_{\mathrm{A}} - k_2 x_{\mathrm{B}} & \dot{x}_{\mathrm{G}} &= k_6 x_{\mathrm{F}} - k_7 x_{\mathrm{G}} \\
\dot{x}_{\mathrm{C}} &= k_2 x_{\mathrm{B}} - k_3 x_{\mathrm{C}} & \dot{x}_{\mathrm{H}} &= k_7 x_{\mathrm{G}} - k_8 x_{\mathrm{H}} \\
\dot{x}_{\mathrm{D}} &= k_3 x_{\mathrm{C}} - k_4 x_{\mathrm{D}} & \dot{x}_{\mathrm{I}} &= k_8 x_{\mathrm{H}} - k_9 x_{\mathrm{I}} \\
\dot{x}_{\mathrm{E}} &= k_4 x_{\mathrm{D}} - k_5 x_{\mathrm{E}} & \dot{x}_{\mathrm{J}} &= k_9 x_{\mathrm{I}}
\end{aligned} \tag{2.58}$$

*Let* $I = [0, 15]$ *s,* $\mathbf{x}_0 = (20, 0, 0, \ldots, 0)$ *M, and assume that all rate coefficients* $\mathbf{k} =$

$(k_1, \ldots, k_9)$ *are only known to within an order of magnitude; i.e.,* $\mathbf{k} \in [\hat{\mathbf{k}}, 10\hat{\mathbf{k}}]$ *with*

$$\hat{\mathbf{k}} = (3.6, 2.4, 4.2, 2.4, 3, 3.6, 4.2, 4.8, 3) \times 10^{-2}.$$

*Given* $\mathbf{x}_0$*, it is easy to see that all concentrations are bounded within* $[0, 20]$ *M. Moreover, this system satisfies the single affine invariant* $\mathbf{1}^{\mathrm{T}}\mathbf{x}(t, \mathbf{k}) = \mathbf{1}^{\mathrm{T}}\mathbf{x}_0$ *stating that the sum of all concentrations remains constant.*

*Examining the right-hand sides of* (2.58) *suggests adding the following eight redundant states:*

$$\begin{aligned}
y_1 &= x_{\mathrm{A}} + x_{\mathrm{B}} \\
y_2 &= x_{\mathrm{A}} + x_{\mathrm{B}} + x_{\mathrm{C}} \\
y_3 &= x_{\mathrm{A}} + x_{\mathrm{B}} + x_{\mathrm{C}} + x_{\mathrm{D}} \\
y_4 &= x_{\mathrm{A}} + x_{\mathrm{B}} + x_{\mathrm{C}} + x_{\mathrm{D}} + x_{\mathrm{E}} \\
y_5 &= x_{\mathrm{A}} + x_{\mathrm{B}} + x_{\mathrm{C}} + x_{\mathrm{D}} + x_{\mathrm{E}} + x_{\mathrm{F}} \\
y_6 &= x_{\mathrm{A}} + x_{\mathrm{B}} + x_{\mathrm{C}} + x_{\mathrm{D}} + x_{\mathrm{E}} + x_{\mathrm{F}} + x_{\mathrm{G}} \\
y_7 &= x_{\mathrm{A}} + x_{\mathrm{B}} + x_{\mathrm{C}} + x_{\mathrm{D}} + x_{\mathrm{E}} + x_{\mathrm{F}} + x_{\mathrm{G}} + x_{\mathrm{H}} \\
y_8 &= x_{\mathrm{A}} + x_{\mathrm{B}} + x_{\mathrm{C}} + x_{\mathrm{D}} + x_{\mathrm{E}} + x_{\mathrm{F}} + x_{\mathrm{G}} + x_{\mathrm{H}} + x_{\mathrm{I}}
\end{aligned} \tag{2.59}$$

*With these definitions, the corresponding ODEs enjoy successive term cancellations,*

*ultimately isolating eight of the reaction rate expressions as follows:*

$$\begin{aligned} \dot{y}_1 &= -k_2 x_{\mathrm{B}} & \dot{y}_5 &= -k_6 x_{\mathrm{F}} \\ \dot{y}_2 &= -k_3 x_{\mathrm{C}} & \dot{y}_6 &= -k_7 x_{\mathrm{G}} \\ \dot{y}_3 &= -k_4 x_{\mathrm{D}} & \dot{y}_7 &= -k_8 x_{\mathrm{H}} \\ \dot{y}_4 &= -k_5 x_{\mathrm{E}} & \dot{y}_8 &= -k_9 x_{\mathrm{I}} \end{aligned} \tag{2.60}$$

*As discussed in §2.4, conservatism can also be reduced by substituting the new state variables into the right-hand side functions for either the original or new states. Thus, we rewrite* (2.60) *equivalently as follows:*

$$\begin{aligned} \dot{y}_1 &= -k_2(y_1 - x_{\mathrm{A}}) & \dot{y}_5 &= -k_6(y_5 - y_4) \\ \dot{y}_2 &= -k_3(y_2 - y_1) & \dot{y}_6 &= -k_7(y_6 - y_5) \\ \dot{y}_3 &= -k_4(y_3 - y_2) & \dot{y}_7 &= -k_8(y_7 - y_6) \\ \dot{y}_4 &= -k_5(y_4 - y_3) & \dot{y}_8 &= -k_9(y_8 - y_7) \end{aligned} \tag{2.61}$$

*Although these equations involve more terms, recall that the interval for $y_i$ will be degenerate in all required interval extensions of $\dot{y}_i$ due to the action of $\mathcal{B}_i^{L/U}$ in* (2.10)*. The final augmented system consists of* (2.58), (2.61)*, and the nine invariants $\mathbf{1}^{\mathrm{T}}\mathbf{x}(t,\mathbf{k}) = \mathbf{1}^{\mathrm{T}}\mathbf{x}_0$ and* (2.59)*.*

*Figure 2.7 shows that the pre-existing invariant in this example does not lead to significant improvement over SDI. In contrast, the use of manufactured invariants achieves very significant improvements. The costs were* 0.002 *s for a single trajectory,* 0.013 *s for SDI,* 0.014 *s for DI using the pre-existing invariant, and* 0.8 *s for DI using both pre-existing and manufactured invariants. Because 8 new states and invariants were added, the improved bounds come at significant additional cost in this*

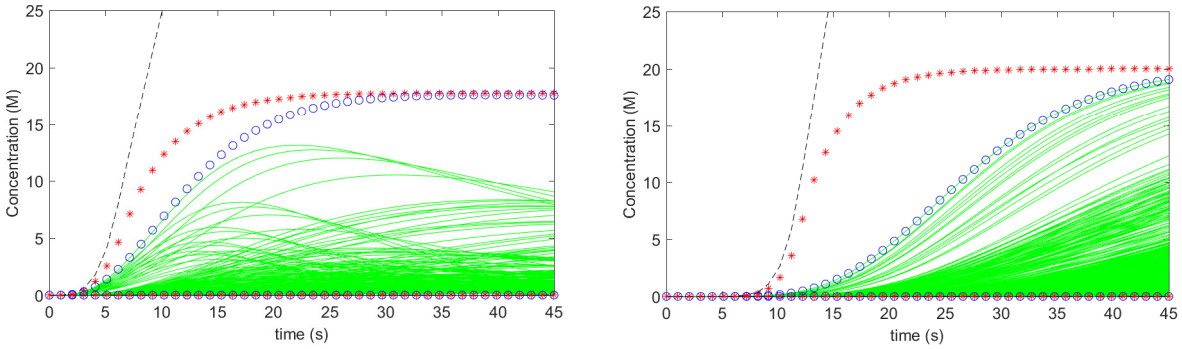


Figure 2.7: State bounds for $x_{\mathrm{E}}$ (left) and $x_{\mathrm{J}}$ (right) in (2.58) computed using SDI (dashed), DI with pre-existing invariants (stars), and DI with both pre-existing and manufactured invariants (circles). Solid lines are real trajectories.

*case. Thus, there is a clear need for heuristics that can identify only the most effective new variables. However, note that* 0.8 *s is only enough time to sample* 400 *trajectories. Considering that this model has* 9 *parameters with an order of magnitude uncertainty in each, this time is quite reasonable for achieving sharp bounds. For example, considering parameter values on a grid with only* 3 *points in each dimension would require* 19683 *trajectories at a cost of 39.4 s.*

**Example 6** *The following model was introduced in [17] and has no pre-existing invariants. It describes a stirred tank reactor with four species:*

$$\begin{aligned}
\dot{x}_{\mathrm{A}} &= -u_3 x_{\mathrm{A}} x_{\mathrm{B}} - k_2 x_{\mathrm{A}} x_{\mathrm{C}} + \tau^{-1}(u_1 - 2x_{\mathrm{A}}) \\
\dot{x}_{\mathrm{B}} &= -u_3 x_{\mathrm{A}} x_{\mathrm{B}} + \tau^{-1}(u_2 - 2x_{\mathrm{B}}) \\
\dot{x}_{\mathrm{C}} &= u_3 x_{\mathrm{A}} x_{\mathrm{B}} - k_2 x_{\mathrm{A}} x_{\mathrm{C}} - 2\tau^{-1} x_{\mathrm{C}} \\
\dot{x}_{\mathrm{D}} &= k_2 x_{\mathrm{A}} x_{\mathrm{C}} - 2\tau^{-1} x_{\mathrm{D}}
\end{aligned} \tag{2.62}$$

*The time horizon is* $I = [0, 15]$ *(s),* $\tau = V/v_A = 20$ *(min),* $k_2 = 0.4$ *($M^{-1}min^{-1}$). There are three uncertain parameters: the inlet concentration of species A,* $u_1 \in$

$[0.9, 1.1]$ *(M), the inlet concentration of species B,* $u_2 \in [0.8, 1.0]$ *(M), and first reaction rate constant* $u_3 \in [10, 50]$ *(*$M^{-1}min^{-1}$*). The initial concentration for all species is zero.*

*In §2.4.1, we described the close relationship between the new bounding approach proposed here and the polyhedral bounding method in [17] when we restrict ourselves to linear combinations of the original states, i.e.,* $\mathbf{y} = \mathbf{g}(\mathbf{x}) = \mathbf{a}^{\mathrm{T}}\mathbf{x}$*. Motivated by this connection, we mimic the results in [17] as closely as possible for this example by defining the following new variables:*

$$
\begin{aligned}
y_1 &= -\frac{1}{3}x_{\mathrm{A}} - \frac{1}{3}x_{\mathrm{B}} + \frac{1}{3}x_{\mathrm{C}} \\
y_2 &= -\frac{1}{3}x_{\mathrm{A}} - \frac{1}{3}x_{\mathrm{C}} + \frac{1}{3}x_{\mathrm{D}} \\
y_3 &= -x_{\mathrm{A}} + 2x_{\mathrm{B}} + x_{\mathrm{C}} \\
y_4 &= x_{\mathrm{A}} - x_{\mathrm{B}} + x_{\mathrm{D}}
\end{aligned}
$$

*With these definition, the corresponding ODEs are:*

$$
\begin{aligned}
\dot{y}_1 &= u_3 x_{\mathrm{A}} x_{\mathrm{B}} - (1/3)\tau^{-1}(u_1 + u_2) - 2\tau^{-1} y_1 \\
\dot{y}_2 &= k_2 x_{\mathrm{A}} x_{\mathrm{C}} - (1/3)\tau^{-1} u_1 - 2\tau^{-1} y_2 \\
\dot{y}_3 &= \tau^{-1}(2(u_2 - y_3) - u_1) \\
\dot{y}_4 &= \tau^{-1}(u_1 - u_2 - 2y_4)
\end{aligned}
\tag{2.63}
$$

*In this case, both methods describe polyhedral enclosures of the reachable set with the same facets. The key difference is that the method in [17] propagates this enclosure forward in time by solving ODEs with linear programs embedded, while the method proposed here uses only interval computations. This is expected to be more efficient,*

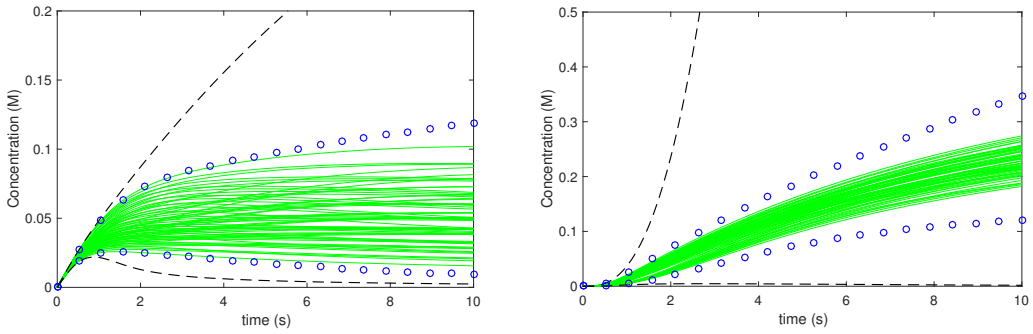


Figure 2.8: State bounds for $x_{\mathrm{A}}$ (left) and $x_{\mathrm{C}}$ (right) in (2.8) computed using SDI (dashed) and DI with manufactured invariants (circles). Solid lines are real trajectories.

*but also implies that we make less effective use of the polyhedral enclosure when bounding the range of each $f_i$ at each time step. However, the new preconditioning scheme outlined in §2.5.1 is designed to minimize this disadvantage.*

*The results are shown in Figure 2.8. Clearly, the use of manufactured invariants results in a very significant improvement over SDI. Moreover, our results are very close to those obtained in [17] using the LP-based polyhedral bounding method, demonstrating that the new preconditioning method is effective. The costs were* 0.003 *s for a single trajectory,* 0.013 *s for SDI, and* 0.10 *s using manufactured invariants. The reported time in [17] is* 0.03 *s using a processor with very similar performance according to the PassMark benchmarks. However, the method in [17] was implemented in* `C++`*. While this makes direct comparison with our* `MATLAB` *implementation difficult,* `C++` *is typically many times faster, implying that our method is competitive at worst, and may be considerably more efficient.*

**Example 7** *The following model describes a two-phase counter-current multistage*

*liquid-liquid extraction system with a single solute [25]:*

$$V_L\dot{x}_n = L(x_{n-1} - x_n) - Q_n \tag{2.64}$$
$$V_G\dot{y}_n = G(y_{n+1} - y_n) + Q_n$$

*Above,* $n = 1, \ldots, 5$ *is the stage number,* $x_n$ *and* $y_n$ *are the concentrations of solute in the feed and solvent phases, respectively (kg/m*$^3$*),* $V_L = 2$ *and* $V_G = 2$ *are the phase volumes (m*$^3$*),* $L = 5$ *and* $G = 5$ *are flow rates (m*$^3$*/h), and* $Q_n$ *is the rate of solute transfer, expressed as*

$$Q_n = K_L a(x_n - x_n^*)V. \tag{2.65}$$

*Above,* $K_L a$ *is the overall mass transfer capacity constant (1/h),* $V = V_L + V_G$ *is the total hold-up volume (m*$^3$*/h), and* $x_n^*$ *is the solute concentration in equilibrium with* $y_n$*. We assume that the following polynomial has been fit to experimental equilibrium data:* $x_n^* = p_1 y_n^4 + p_2 y_n^3 + p_3 y_n^2 + p_4 y_n + p_5$*. Due to measurement error, we further assume that* $K_L a$ *and all coefficient* $p_1, ..., p_5$ *are uncertain, with* $K_L a \in [8, 16]$*,* $p_1 \in [1.48, 1.49] \times 10^{-5}$*,* $p_2 \in [-1.11, -1.05] \times 10^{-3}$*,* $p_3 \in [3.28, 3.30] \times 10^{-3}$*,* $p_4 \in [7.56, 7.58] \times 10^{-1}$*, and* $p_5 \in [4.93, 4.95] \times 10^{-2}$*. All initial concentrations are* 0 *and the inlet flow rates are* $x_0 = 10$ *and* $y_0 = 1$ *(m*$^3$*/h).*

*Observing that* $Q_n$ *appears in both ODEs in* (2.64)*, effective redundant state variables can be created by arranging for the cancellation of this term. Specifically, we define*

$$N_n = V_L x_n + V_G y_n, \quad n = 1, \ldots, 5, \tag{2.66}$$

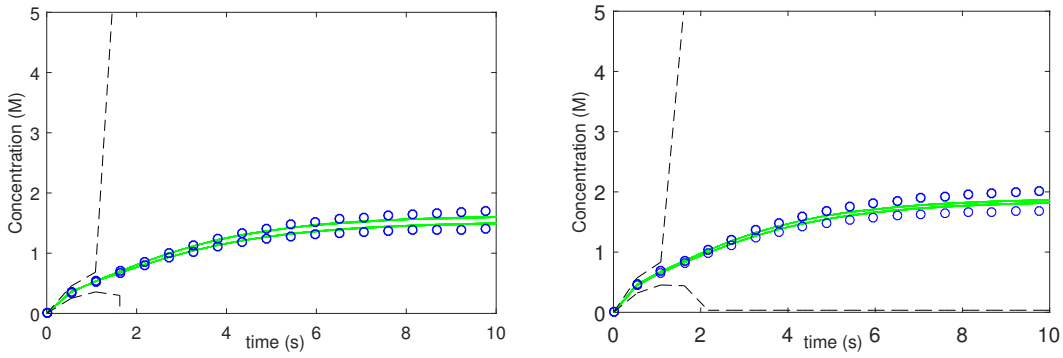


Figure 2.9: State bounds for $x_5$ (left) and $y_5$ (right) from (2.64) computed using SDI (dashed) and DI with manufactured invariants (circles). Solid lines are real trajectories

*which leads to the augmented ODEs*

$$\dot{N}_n = Lx_{n-1} + Gy_{n+1} - (Lx_n + Gy_n), \tag{2.67}$$

$$= 5(x_{n-1} + y_{n+1}) - \frac{5}{2}N_n, \quad n = 1, \ldots, 5. \tag{2.68}$$

*Figure 2.9 clearly shows that the SDI bounds rapidly explode for this example. In contrast, the use of manufactured model redundancy provides bounds that are nearly exact. The costs were* 0.04 *s for a single trajectory, and* 0.7 *s for DI using manufactured invariants, while integration of the SDI bounds failed due to rapid divergence around* $t = 2$.

**Example 8** *The Van der Pol equations describe an oscillator with applications in electrical circuits, biological networks, and various other domains. We study these equations here because oscillatory systems are notoriously difficult to bound. The Van der Pol equations in two-dimensions are*

$$\dot{x}_1 = x_2, \qquad \dot{x}_2 = (1 - {x_1}^2)x_2 - x_1. \tag{2.69}$$

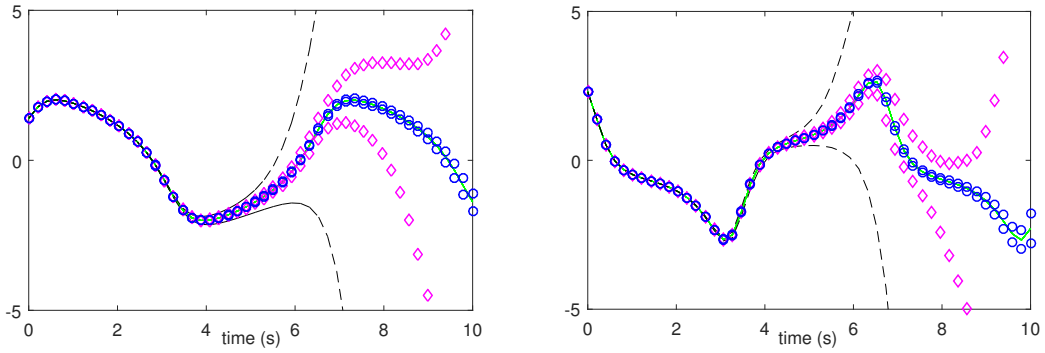


Figure 2.10: State bounds for $x_1$ (left) and $x_2$ (right) in (2.69) computed using SDI (dashed), DI with manufactured state variable $y_1$ only (diamond), and DI with both manufactured state variables $y_1$ and $y_2$ (circle). Solid lines are real trajectories.

*The time horizon is* $I = [0, 10]$ *(s) and the initial conditions are uncertain with* $x_1(t_0) \in [1.399, 1.400]$ *and* $x_2(t_0) \in [2.299, 2.300]$. *To endow this system with affine solution invariants, we define the redundant states* $y_1 \equiv x_1 - x_2$ *and* $y_2 \equiv x_1 + x_2$, *and, after some algebraic rearrangements, augment* (2.69) *with the additional ODEs*

$$\dot{y}_1 = x_1(x_1x_2 + 1), \qquad \dot{y}_2 = (1 - x_1^2)x_2 - y_1. \tag{2.70}$$

*Figure 2.10 compares the results of applying SDI to* (2.69) *and applying DI with manufactured invariants to* (2.69)–(2.70). *The figure also shows the result of adding only a single invariant and the single additional state* $y_1$. *Again, the simple addition of linear invariants leads to a dramatic reduction in the conservatism of the DI approach. Because the Van der Pol system is oscillatory and highly sensitive to the initial conditions, it is expected that the bounds computed by any method will eventually diverge. However, the use of redundancy here has allowed DI to produce meaningful bounds over a significantly longer time horizon. The costs were* 0.005 *for single trajectory,* 0.018 *s for SDI,* 0.05 *s using one manufactured invariant, and* 0.08 *s for using two manufactured invariants.*

**Example 9** *The Lotka-Volterra system is widely used for comparing bounding algorithms and has been studied in several previous articles [17, 35]. For comparison, we use the same data here. The Lotka-Volterra equations are*

$$\dot{x}_1 = u_1 x_1(1 - x_2), \qquad \dot{x}_2 = u_2 x_2(x_1 - 1). \tag{2.71}$$

*The time horizon is* $I = [0, 10]$ *(s) and* $(u_1, u_2)$ *is uncertain with* $u_1 \in [2.99, 3.01]$ *and* $u_2 \in [0.99, 1.01]$*. The initial condition is* $(x_1, x_2)(t_0) = (1.2, 1.1)$*.*

*We define the following redundant state variables:*

$$\begin{aligned}
y_1 &= \frac{x_1}{u_1} - \frac{x_2}{u_2} \\
y_2 &= x_1 x_2 \\
y_3 &= \frac{x_1}{x_2} \\
y_4 &= \sin(\pi/16)x_1 + \cos(\pi/16)x_2 \\
y_5 &= \sin(2\pi/16)x_1 + \cos(2\pi/16)x_2 \\
y_6 &= \sin(3\pi/16)x_1 + \cos(3\pi/16)x_2 \\
y_7 &= \sin(5\pi/16)x_1 + \cos(5\pi/16)x_2 \\
y_8 &= \sin(7\pi/16)x_1 + \cos(7\pi/16)x_2
\end{aligned} \tag{2.72}$$

*As in Example 6, the linear invariants defining* $y_4$*–*$y_8$ *are chosen to mimic the polyhedral enclosure used in [17], as described in §2.4.1. However, in contrast to our previous examples, we also consider three* nonlinear *definitions. Note that these are permissible in Theorem 4, provided that an appropriate* $\mathcal{I}_G$ *satisfying Assumption 2 is defined. In contrast, the polyhedral bounding method proposed in [17] can only make use of linear constraints.*

*To satisfy Assumption 2, we define a custom interval refinement mapping called $\mathcal{N}_G$ as follows. First, given intervals $X$, $Y$, and $U$, the three nonlinear invariants in* (2.72) *are used the refine $X$ and $Y$ by taking interval extensions of rearrangements of these equations in the following order:*

$$
\begin{aligned}
X_1 &:= X_1 \cap [U_1(Y_1 + X_2/U_2)] \\
X_1 &:= X_1 \cap [Y_2/X_2] \\
X_1 &:= X_1 \cap [Y_3 X_2] \\
X_2 &:= X_2 \cap [U_2(X_1/U_1 - Y_1)] \\
X_2 &:= X_2 \cap [Y_2/X_1] \\
X_2 &:= X_2 \cap [X_1/Y_3] \\
Y_1 &:= Y_1 \cap [X_1/U_1 - X_2/U_2] \\
Y_2 &:= Y_2 \cap [X_1 X_2] \\
Y_3 &:= Y_3 \cap [X_1/X_2]
\end{aligned}
\tag{2.73}
$$

*After these refinements, the resulting intervals are passed to $\mathcal{I}_G$ as defined in Algorithm 2 using the remaining linear invariants defining $y_4$–$y_8$ in* (2.72)*. The overall operation satisfies Assumption 2 due to the Lipschitz continuity of interval arithmetic, provided that none of the intersections are empty. To guard against this, we use a simple Lipschitz extension of the intersection defined and discussed in detail in [61].*

*The results are shown in Figure 2.11. Again, the addition of manufactured invariants results in much tighter bounds than SDI. In fact, the resulting bounds are extremely similar to those obtained using the LP-based polyhedral bounding algorithm in [17]. The costs were* 0.003 *s for single trajectory,* 0.016 *s for SDI, and* 0.10 *s for DI using manufactured model redundancy. The time reported in [17] is* 0.050 *s.*

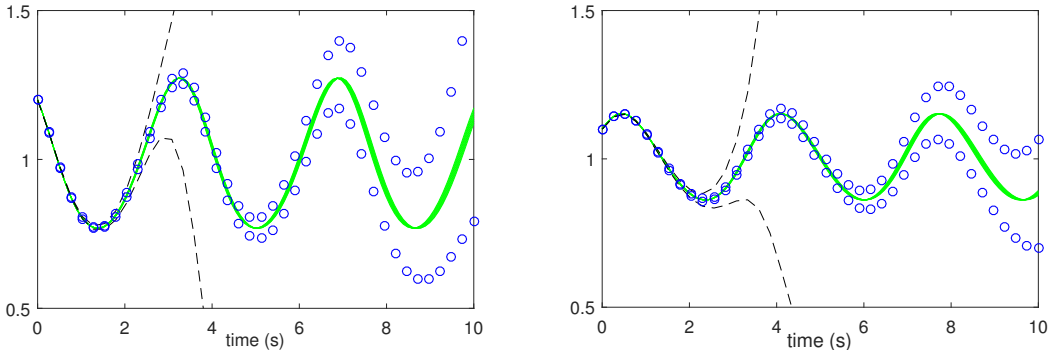


Figure 2.11: State bounds for $x_1$ (left)and $x_2$ (right) computed from computed using SDI (dashed), and DI with manufactured affine invariants under mapping $\mathcal{N}_G$ (circle), solid line presents the real trajectories.

*However, we note again that the implementation in [17] is in* `C++`*, which is expected to be considerably faster than an equivalent implementation in* `MATLAB`*. Thus, this example shows that similar quality bounds can be obtained using only interval methods, likely with significant gains in efficiency.*

## 2.7 Conclusion

This chapter presented a novel strategy for reducing the conservatism of fast interval methods for bounding the solutions of nonlinear ODEs through the addition of redundant state variables and solution invariants. We have outlined several mechanisms through which these redundant equations can improve state bounds relative to standard interval methods, and presented preconditioning heuristics for effectively using redundant solution invariants in interval refinement algorithms. Our numerical results clearly illustrate the ability of model redundancy to reduce conservatism, often dramatically, at modest additional cost. However, several critical issues remain be addressed, including the automatic generation of effective redundant equations, and the use of general nonlinear solution invariants.

# Chapter 3

# Exploiting Nonlinear Invariants and Path Constraints to Achieve Tighter Bounds on the Flows of Uncertain Nonlinear Systems using Differential Inequalities

## 3.1 Introduction

This chapter presents a new method for computing guaranteed bounds on the solutions of systems of nonlinear ordinary differential equations (ODEs) subject to uncertain initial conditions, parameters, and time-varying inputs. Such bounds are widely used in algorithms for robust state estimation [54, 42], set-based fault detection [60, 53, 35], system verification [3, 8, 28], robust control [78, 32, 15], and solving open-loop optimal control problems to guaranteed global optimality [62, 22]. Ac-

cordingly, a wide variety of methods have been proposed for computing such bounds, including simulation-based methods [26, 37], level-set methods [41, 30], conservative linearization methods [5], validated integration methods based on interval Taylor series expansions and Taylor models [47, 39, 34, 24], and methods based on the theory of differential inequalities [64, 77, 67]. However, for nonlinear systems with large uncertainties, obtaining guaranteed bounds that are both accurate and computationally efficient remains a significant challenge.

This chapter continues a line of research that has recently achieved very promising bounding results using methods based on differential inequalities (DI). DI methods provide rigorous interval or polyhedral bounds at very low computational cost relative to alternative approaches [64, 17], making them particularly promising for online control applications such as state estimation and safety verification [54, 42, 3, 8], as well as for use in branch-and-bound algorithms for solving optimal control problems to global optimality [62, 22]. Although the original DI method in [16] often produces extremely conservative bounds, several DI methods with greatly improved accuracy have since been developed, specifically for systems whose states are known to satisfy a set of state constraints pointwise in time [74, 58, 64, 17, 19, 67]. Examples of such constraints include physically motivated upper and lower bounds, such as nonnegativity of certain states [74], or algebraic functions of the states that are known to remain constant with time due to, e.g., the conservation of mass, energy, or chemical elements (we call such functions *solution invariants*) [64]. Constraints of this type are implied by the dynamics, and are therefore satisfied by all system trajectories. In contrast, the article [19] considers state constraints that are externally imposed, such as path constraints in the context of dynamic optimization, where one is only interested in bounding the feasible trajectories. In both cases, enhanced DI methods have been developed that can exploit these constraints during the bounding

procedure, often resulting in much tighter bounds with only a moderate increase in computational cost [64, 17, 19]. In brief, this is accomplished by applying a suitably defined bound refinement operator pointwise in time during the forward propagation of the bounds. At each point in time, this refinement operator attempts to shrink the current bounds by eliminating enclosed regions that violate the constraints. Until recently, an evident drawback of these DI methods is that they only apply to systems for which appropriate constraints are known *a priori*. However, the article [67] describes a framework for applying these methods to arbitrary, unconstrained nonlinear ODEs by introducing redundant state variables and ODEs, which effectively embeds the ODEs of interest into a higher-dimensional system that obeys a set of solution invariants by design. Numerous case studies in [67] show that this is very effective, although it requires significant problem specific insight in most cases.

Despite these recent advances, the DI methods described above suffer from some significant limitations, which are discussed in turn in the following paragraphs. In brief, the main contribution of this chapter is to overcome several of these limitations by establishing a new, more general DI theorem, and a new bounding algorithm based on this theorem. The specific new capabilities that result from these developments are discussed in detail below.

*Nonlinear constraints.* With one problem-specific exception in [67], none of the existing DI algorithms are applicable to problems with nonlinear constraints. This is a significant limitation because many systems of interest naturally obey nonlinear solution invariants that could be very helpful for achieving tighter bounds on their solutions (e.g., oscillators and Hamiltonian systems [71]). Moreover, nonlinear path constraints arise in a wide variety of optimal control problems [13]. In principle, the key theorems underlying state-of-the-art DI bounding methods do permit the use of nonlinear constraints [64, 19]. However, all of these methods are implemented using

refinement operators that must satisfy certain theoretical conditions that prove to be very restrictive in the nonlinear case. Specifically, the standard assumption on the domain of this operator implies that the nonlinear constraint functions must be well-defined for arbitrary values of their arguments, which is often not the case (this issue is discussed further in §3.4). Moreover, the refinement operator is required to be locally Lipschitz continuous, which is not true of most standard algorithms for refining bounds based on a set of nonlinear constraints [48]. To address the first limitation, our new DI theorem uses an alternative assumption on the domain of the refinement operator. We show that this assumption is easily verifiable for a very general class of nonlinear constraints. In addition, we present a new refinement algorithm for this class of constraints that is guaranteed to satisfy the required Lipschitz property. Together, these contributions result in an new DI bounding algorithm that can effectively exploit very general nonlinear constraints in order to achieve tighter bounds.

*Constraints depending on uncertain time-invariant parameters and time-varying inputs.* The key theorems underlying the state-of-the-art DI bounding methods in [64, 19] only permit the use of constraints that depend exclusively on the system states. Specifically, these constraints cannot depend on uncertain model parameters or inputs. In the article [19], it was observed that constraints depending on time-invariant uncertain parameters are permissible if these parameters are regarded as additional states with uncertain initial conditions and zero time derivatives. However, dependence on time-varying inputs was not addressed there. These are significant limitations because many systems of practical interest obey solution invariants that depend on uncertain parameters and inputs. One specific example is the Lotka-Voterra oscillator, which is discussed further in §3.6. Moreover, joint input-state constraints are common in many optimal control formulations, including robot mo-

tion planning and flight path planning problems [31, 13]. To address these limitations, our new DI theorem permits constraints with arbitrary dependence on both uncertain time-invariant parameters and uncertain time-varying inputs.

*Constraints depending on time derivatives of the states.* Existing DI methods that exploit state constraints cannot be applied to constraints that depend on the time-derivatives of the state variables. However, such constraints arise naturally in many applications (e.g., non-holonomic robot dynamics [31]), and solution invariants for many systems are most easily formulated in terms of state derivatives. To address this limitation, both the new DI theorem presented here and its algorithmic implementation support the use of constraints with arbitrary dependence the time derivatives of the system states.

In aggregate, the theoretical and algorithmic contributions outlined above significantly increase the applicability of state-of-the-art DI methods for computing sharp bounds on the solutions of uncertain nonlinear systems. Specifically, our results enable such methods to address a wide variety of systems with state constraints that are nonlinear and potentially dependent on uncertain model parameters, inputs, and state derivatives.

The new theory and algorithms presented for constrained ODEs in this chapter are similar in many respects to the results for bounding the solutions of semi-explicit differential-algebraic equations (DAEs) in [59, 61]. However, these problems are distinct because the ODE systems considered here are uniquely solvable without constraints. In other words, the constraints are used here only as additional information to improve the computed bounds. In contrast, the algebraic equations in the semi-explicit DAEs in [59, 61] are necessary for specifying locally unique solutions. Importantly, the DAE bounding theory in [59] and the properties of the refinement operator developed in [61] both fundamentally rely on uniqueness conditions for the

algebraic equations that would be unreasonable to assume for the constraints considered in this chapter. Thus, the key contributions of this chapter are distinct from those in [59, 61].

## 3.2 Preliminary Notation and Definitions

Let $\bar{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$ denote the extended real line. For any measurable $I \subset \mathbb{R}$, denote the space of Lebesgue integrable functions $u : I \to \bar{\mathbb{R}}$ by $L^1(I)$. A vector function $\mathbf{u} : I \to \bar{\mathbb{R}}^n$ is Lebesgue integrable if $u_i \in L^1(I)$ for each $i = 1, \ldots, n$, in which case we write $\mathbf{u} \in (L^1(I))^n$. Furthermore, let $\mathcal{AC}(I, \mathbb{R}^n)$ denote the space of absolutely continuous functions from $I$ into $\mathbb{R}^n$.

For $\mathbf{v}, \mathbf{w} \in \mathbb{R}^n$, let $[\mathbf{v}, \mathbf{w}]$ denote the compact $n$-dimensional interval $\{\mathbf{z} \in \mathbb{R}^n : \mathbf{v} \leq \mathbf{z} \leq \mathbf{w}\}$. The set of all nonempty interval subsets of $\mathbb{R}^n$ is denoted by $\mathbb{IR}^n$. Similarly, for $D \subset \mathbb{R}^n$, the set of all nonempty interval subsets of $D$ is denoted by $\mathbb{I}D$. Let $D \subset \mathbb{R}^n$ and $\mathbf{f} : D \to \mathbb{R}^m$. A mapping $F : \mathcal{D} \subset \mathbb{I}D \to \mathbb{IR}^m$ is an *inclusion function* for $\mathbf{f}$ on $\mathcal{D}$ if

$$F(X) \supset \mathbf{f}(X) \equiv \{\mathbf{f}(\mathbf{x}) : \mathbf{x} \in X\}, \quad \forall X \in \mathcal{D}. \tag{3.1}$$

Inclusion functions can be readily derived for a very general class of nonlinear functions called *factorable functions*. In brief, $\mathbf{f}$ is factorable if it can be evaluated by a finite recursive composition of binary additions, binary multiplications, and standard univariate functions such as $-x$, $\frac{1}{x}$, $x^n$, $e^x$, etc. This includes nearly every function that can be written explicitly in computer code. For any factorable function $\mathbf{f}$, a specific inclusion function called the *natural interval extension* can be constructed by simply replacing each operation in the definition of $\mathbf{f}$ with a suitable interval coun-

terpart [48]. In the following sections, we make use of several properties of natural interval extensions from [57].

It is well known that the Hausdorff distance $d_H$ (induced by $\|\cdot\|_\infty$) is a metric on $\mathbb{IR}^n$. Therefore, standard definitions and results concerning sets and functions on metric spaces are applicable. For example, the open ball of radius $\eta > 0$ centered at $X \in \mathbb{IR}^n$ is defined by $B_\eta(X) \equiv \{Z \in \mathbb{IR}^n : d_H(X, Z) < \eta\}$. Similarly, a set $\mathcal{X} \subset \mathbb{IR}^n$ (i.e., $\mathcal{X}$ is a set whose elements are intervals) is called open if for every $X \in \mathcal{X}$, $\exists \eta > 0$ such that $B_\eta(X) \subset \mathcal{X}$. Additionally, a mapping $F : \mathcal{D} \subset \mathbb{IR}^n \to \mathbb{IR}^m$ is called locally Lipschitz continuous on $\mathcal{D}$ if for every $X \in \mathcal{D}$, $\exists L, \eta > 0$ such that $d_H(F(\bar{X}), F(\hat{X})) \leq L d_H(\bar{X}, \hat{X})$ for every $\bar{X}, \hat{X} \in B_\eta(X) \cap \mathcal{D}$. These definitions will be used in several places in conjunction with the standard facts that the pre-image of an open set under a continuous function is open, and that the composition of locally Lipschitz functions is locally Lipschitz, both of which hold in general metric spaces. For brevity in the remainder of the chapter, we use $\|\cdot\|$ to denote the infinity norm $\|\cdot\|_\infty$.

## 3.3 Problem Statement

Let $\mathbf{f} : D_f \subset \mathbb{R} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ be a vector field, let $G \subset \mathbb{R} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_x}$ be a constraint set, and consider the constrained dynamic system

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{u}(t), \mathbf{x}(t)), \tag{3.2a}$$

$$\mathbf{x}(t_0) = \mathbf{x}_0, \tag{3.2b}$$

$$(t, \mathbf{u}(t), \mathbf{x}(t), \dot{\mathbf{x}}(t)) \in G. \tag{3.2c}$$

Let $I = [t_0, t_f] \subset \mathbb{R}$ be a time horizon of interest, let $U \in \mathbb{IR}^{n_u}$, and define the set of admissible inputs as

$$\mathcal{U} \equiv \{\mathbf{u} \in (L^1(I))^{n_u} : \mathbf{u}(t) \in U \text{ for almost every (a.e.) } t \in I\}.$$

Moreover, let $X_0 \in \mathbb{IR}^{n_x}$ be an interval of admissible initial conditions. For any fixed $(\mathbf{x}_0, \mathbf{u}) \in X_0 \times \mathcal{U}$, we call $\mathbf{x} \in \mathcal{AC}(I, \mathbb{R}^{n_x})$ a solution of (4.1a)–(4.1b) if it satisfies (4.1a)–(4.1b) for a.e. $t \in I$. Considering the constraint (4.1c) as well, we call the triple $(\mathbf{x}_0, \mathbf{u}, \mathbf{x}) \in X_0 \times \mathcal{U} \times \mathcal{AC}(I, \mathbb{R}^{n_x})$ a solution of (4.1) if it satisfies (4.1a)–(4.1c) for a.e. $t \in I$.

We assume throughout that a unique solution of (4.1a)–(4.1b) exists for every choice of $(\mathbf{x}_0, \mathbf{u}) \in X_0 \times \mathcal{U}$ and, when necessary for clarity, we denote this solution by $\mathbf{x}(t; \mathbf{x}_0, \mathbf{u})$. Existence and uniqueness can be ensured locally under standard assumptions on $\mathbf{f}$. Notably, these include a Lipschitz condition that plays a central role in the bounding theories in [64, 67, 17, 19]. In contrast, our main result here does not require any specific assumptions on $\mathbf{f}$ other than existence and uniqueness of the solution of (4.1a)–(4.1b). Likewise, we put no restrictions on the constraint set $G$. However, our main result does require the existence of a locally Lipschitz interval function $\mathcal{R}$ satisfying a certain inclusion property with respect to $\mathbf{f}$ and $G$, which is assumed explicitly in §3.5.

**Definition 8** *The* reachable set *of the constrained system* (4.1) *is defined for every* $t \in I$ *as*

$$Re(t) \equiv \{\mathbf{z} \in \mathbb{R}^{n_x} : \exists (\mathbf{x}_0, \mathbf{u}, \mathbf{x}) \in X_0 \times \mathcal{U} \times \mathcal{AC}(I, \mathbb{R}^{n_x}) \text{ of } (4.1) \text{ satisfying } \mathbf{x}(t) = \mathbf{z}\}. \tag{3.3}$$

We are interested in computing a tight, time-varying enclosure of the reachable set of (4.1). In other words, we wish to bound all possible solutions of (4.1a)–(4.1b) that have uncertain initial conditions and inputs in $X_0 \times \mathcal{U}$, and that satisfy the state constraint (4.1c). Depending on the application, the uncertain input $\mathbf{u}$ may represent process disturbances, possible control inputs, unknown model parameters, unmodeled dynamics, etc. Note that the case where some elements of $\mathbf{u}$ are time-invariant parameters taking values in $U$ is a special case of our general formulation in the sense that any valid enclosure of the reachable set of (4.1) also necessarily encloses the smaller set of states that are reachable when some elements of $\mathbf{u}$ are required to be time-invariant.

This chapter specifically considers fast interval methods based on differential inequalities (DI) [64], which furnish *state bounds* defined as follows.

**Definition 9** *Two functions* $\mathbf{v}, \mathbf{w} \in \mathcal{AC}(I, \mathbb{R}^{n_x})$ *are called* state bounds *for* (4.1) *if* $Re(t) \subset [\mathbf{v}(t), \mathbf{w}(t)]$, $\forall t \in I$.

## 3.4 A General State Bounding Theorem for Constrained ODEs

This section presents the main theoretical result of this chapter, Theorem 7, which provides sufficient conditions for a time-varying interval to describe valid state bounds for the constrained system (4.1). This result relies on an interval-valued function $\mathcal{R}$ that must satisfy several key requirements enumerated in Assumption 3. In this assumption and elsewhere, $\mathbf{p}$ is used to denote an arbitrary vector in $U$ so that $\mathbf{u}$ can consistently denote a function in $\mathcal{U}$. Moreover, $P$ is used to denote an arbitrary interval subset of $\mathbb{R}^{n_u}$. Similarly, $\mathbf{z}$ and $\boldsymbol{\sigma}$ are used to denote arbitrary vectors in

$\mathbb{R}^{n_x}$ so that $\mathbf{x}$ and $\dot{\mathbf{x}}$ can consistently denote a solution of (4.1a)–(4.1c) and its time derivative, and $Z$ and $\Sigma$ are used to denote arbitrary interval subsets of $\mathbb{R}^{n_x}$.

To describe the required function $\mathcal{R}$ conceptually, consider an interval $P \times Z$ that encloses all $(\mathbf{u}(t), \mathbf{x}(t))$ corresponding to solutions of (4.1) at some $t \in I$. Assumption 3 below essentially states that an interval function $\mathcal{R}$ is available that takes $(t, P, Z)$ as input and produces an enclosure $\Sigma$ of all possible values of $\boldsymbol{\sigma} = \mathbf{f}(t, \mathbf{p}, \mathbf{z})$ such that $(\mathbf{p}, \mathbf{z}) \in P \times Z$ and $(t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) \in G$. Thus, $\mathcal{R}$ is like an inclusion function for $\mathbf{f}$, but differs in that it is only required to bound $\mathbf{f}$ over arguments that satisfy the constraint set $G$. Assumption 3 further requires $\mathcal{R}$ to satisfy several technical conditions used in the proof of Theorem 7. A specific algorithm for evaluating $\mathcal{R}$ in a manner that satisfies all of these requirements is developed in §3.5.

**Assumption 3** *Let $\mathcal{R} : D_{\mathcal{R}} \subset \mathbb{R} \times \mathbb{IR}^{n_u} \times \mathbb{IR}^{n_x} \to \mathbb{IR}^{n_x}$ be an interval function with the following properties:*

1. *For every $(t, P, Z) \in D_{\mathcal{R}}$, the set $\{t\} \times P \times Z$ is contained in the domain of $\mathbf{f}$, $D_f$, and*

$$\mathcal{R}(t, P, Z) \supset \{\boldsymbol{\sigma} \in \mathbb{R}^{n_x} : \boldsymbol{\sigma} = \mathbf{f}(t, \mathbf{p}, \mathbf{z}),\ (t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) \in G,\ (\mathbf{p}, \mathbf{z}) \in P \times Z\}. \tag{3.4}$$

2. *$D_{\mathcal{R}}$ is open with respect to $t$ and $Z$. Specifically, for every $(\hat{t}, \hat{P}, \hat{Z}) \in D_{\mathcal{R}}$, there exists $\eta > 0$ such that $(t, \hat{P}, Z) \in D_{\mathcal{R}}$ for every $t \in B_\eta(\hat{t})$ and $Z \in B_\eta(\hat{Z})$.*

3. *$\mathcal{R}$ is locally Lipschitz continuous with respect to $Z$, uniformly with respect to $t$. Specifically, for any $(\hat{t}, \hat{P}, \hat{Z}) \in D_{\mathcal{R}}$, there exists $\eta, L > 0$ such that*

$$d_H(\mathcal{R}(t, \hat{P}, Z), \mathcal{R}(t, \hat{P}, \tilde{Z})) \leq L d_H(Z, \tilde{Z}), \tag{3.5}$$

*for every* $t \in B_\eta(\hat{t})$ *and* $Z, \tilde{Z} \in B_\eta(\hat{Z})$.

Our main result below uses $\mathcal{R}$ to formulate a set of differential inequalities that ensure that a time-varying interval $X(t) = [\mathbf{x}^L(t), \mathbf{x}^U(t)]$ contains the reachable set of (4.1) pointwise in time. As with previous bounding results of this type, our result relies on the key idea that, in order for a solution $\mathbf{x}(t)$ to leave the bounds $X(t)$, it must cross the boundary of $X(t)$. More specifically, there must be some $\hat{t} \in I$ at which either $x_i(\hat{t}) = x_i^L(\hat{t})$ or $x_i(\hat{t}) = x_i^U(\hat{t})$ for some $i \in \{1, \ldots, n_x\}$. As a consequence, sufficient conditions for $\mathbf{x}(t)$ to remain in $X(t)$ can be formulated only in terms of the values of the vector field $\mathbf{f}$ on the facets of $X(t)$. Definition 15 provides convenient notation for these facets.

**Definition 10** *Define* $\mathcal{B}_i^L, \mathcal{B}_i^U : \mathbb{IR}^{n_x} \to \mathbb{IR}^{n_x}$ *for every* $i \in \{1, \ldots, n_x\}$ *and every* $X = [\mathbf{x}^L, \mathbf{x}^U] \in \mathbb{IR}^{n_x}$ *by* $\mathcal{B}_i^L(X) = \{\mathbf{z} \in X : z_i = x_i^L\}$ *and* $\mathcal{B}_i^U(X) = \{\mathbf{z} \in X : z_i = x_i^U\}$.

Now, consider a time-varying interval $X(t) = [\mathbf{x}^L(t), \mathbf{x}^U(t)]$ such that $X(t_0) \supset X_0$. Omitting some technical details, the standard differential inequalities result for unconstrained systems [16] states that $\mathbf{x}^L$ and $\mathbf{x}^U$ are valid state bounds if, for every $i \in \{1, \ldots, n_x\}$ and a.e. $t \in I$, they satisfy

$$\dot{x}_i^L(t) \leq f_i(t, \mathbf{p}, \mathbf{z}), \quad \forall (\mathbf{p}, \mathbf{z}) \in U \times \mathcal{B}_i^L(X(t)) \tag{3.6}$$

$$\dot{x}_i^U(t) \geq f_i(t, \mathbf{p}, \mathbf{z}), \quad \forall (\mathbf{p}, \mathbf{z}) \in U \times \mathcal{B}_i^U(X(t)). \tag{3.7}$$

According to the discussion above, the purpose of the first inequality is to ensure that, if any solution of (4.1) is incident on the $i^{\text{th}}$ lower bound at $t$ (i.e., $\mathbf{x}(t) \in \mathcal{B}_i^L(X(t))$), then $x_i^L$ is decreasing faster than $x_i$ at $t$. Of course, the purpose of the second inequality is analogous. Our new result weakens these differential inequalities by

exploiting the state constraint (4.1c) through the operator $\mathcal{R}$. For example, the differential inequality for $x_i^L$ is replaced by

$$\dot{x}_i^L(t) \leq \sigma_i, \quad \forall \boldsymbol{\sigma} \in \mathcal{R}[t, U, \mathcal{B}_i^L(X(t))]. \tag{3.8}$$

By Condition 1 of Assumption 3, this requires $x_i^L(t)$ to be less than $f_i(t, \mathbf{p}, \mathbf{z})$ for all arguments that satisfy $(\mathbf{p}, \mathbf{z}) \in U \times \mathcal{B}_i^L(X(t))$ and the constraint $(t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) \in G$ with $\boldsymbol{\sigma} = \mathbf{f}(t, \mathbf{p}, \mathbf{z})$. The sufficiency of these modified differential inequalities is established in Theorem 7 below. In the stated hypotheses, the shorthand $\mathcal{B}_i^{L/U}$ indicates that a hypothesis must hold with both $\mathcal{B}_i^L$ and $\mathcal{B}_i^U$ independently.

**Theorem 7** *Let* $\mathbf{x}^L, \mathbf{x}^U \in \mathcal{AC}(I, \mathbb{R}^{n_x})$ *and denote* $X(t) \equiv [\mathbf{x}^L(t), \mathbf{x}^U(t)]$, $\forall t \in I$. *Assume that* $X$ *satisfies:*

1. $[t, U, \mathcal{B}_i^{L/U}(X(t))] \in D_{\mathcal{R}}$ *for all* $t \in I$ *and every* $i \in \{1, \ldots, n_x\}$.
2. $X(t_0) \supset X_0$.
3. *For a.e.* $t \in I$ *and every* $i \in \{1, \ldots, n_x\}$,
   (a) $\dot{x}_i^L(t) \leq \sigma_i$ *for all* $\boldsymbol{\sigma} \in \mathcal{R}[t, U, \mathcal{B}_i^L(X(t))]$,
   (b) $\dot{x}_i^U(t) \geq \sigma_i$ *for all* $\boldsymbol{\sigma} \in \mathcal{R}[t, U, \mathcal{B}_i^U(X(t))]$.

*Then, every solution* $(\mathbf{x}_0, \mathbf{u}, \mathbf{x}) \in X_0 \times \mathcal{U} \times \mathcal{AC}(I, \mathbb{R}^{n_x})$ *of* (4.1) *satisfies* $\mathbf{x}(t) \in X(t)$, $\forall t \in I$.

**Proof** See Appendix. □

In Theorem 7, Condition 1 simply ensures that $\mathcal{R}$ is well-defined for the arguments used in Condition 3. Condition 2 requires that $X(t_0)$ bounds all initial

conditions that are admissible in (4.1). Finally, the differential inequalities in Condition 3 ensure that, if some solution of (4.1) is incident on a facet of $X(t)$, then the upper (resp. lower) bound in question is increasing (resp. decreasing) at least as quickly as the corresponding component of the offending solution. Although the purpose of each of these hypotheses is clear, the proof of Theorem 7 is technical and relies on some cumbersome preliminary results. Thus, the formal argument is presented in the Appendix A.

Theorem 7 is similar to several previously published results, with the most closely related being Theorem 2 in [64] (with Definition 3 and Eq. (7)), Theorem 1 in [17], and Theorem 1 in [19]. However, Theorem 7 here is unique in a few key respects. First, prior results have not considered systems with constraints that depend on time-varying uncertain inputs $\mathbf{u}$ or state derivatives $\dot{\mathbf{x}}$, which are permitted in (4.1c) here. Second, there are technical differences in the assumptions required on the domain of $\mathcal{R}$ here relative to previous results, and these are important for effectively exploiting nonlinear constraints. Prior uses of interval operators similar to $\mathcal{R}$ in differential inequalities theorems are based on the theory originally developed in [64]. There, the authors considered state constraints of the form $\mathbf{x}(t) \in G \subset \mathbb{R}^{n_x}$ and used an interval-valued operator of the form $\mathcal{I}_G : D_{\mathcal{I}} \subset \mathbb{IR}^{n_x} \to \mathbb{IR}^{n_x}$ that refines a given interval $Z \in \mathbb{IR}^{n_x}$ by producing an interval enclosure of $Z \cap G$. Critically, this theory requires the strong assumption that every $Z \in \mathbb{IR}^{n_x}$ satisfying $Z \cap G \neq \emptyset$ is an element of $D_{\mathcal{I}}$ (i.e., $\mathcal{I}_G(Z)$ must be well-defined for any such interval). The extensions of this theory in [17] and [19] consider polyhedral rather than interval state bounds, but impose directly analogous forms of this same assumption. Notably, it follows from this assumption that $D_{\mathcal{I}}$ must contain intervals that are arbitrarily large in width, provided that they contain $G$. This assumption is not particularly restrictive when $G$ is defined by a system of linear constraints, and several suitable refinement operators

have been proposed for this case [64, 18, 17, 19, 67]. However, we argue in the next section that this assumption is unreasonable when $G$ is defined more generally by a set of nonlinear constraints (see Remark 1). In contrast, Theorem 7 here does not require any assumption of this type on $\mathcal{R}$. To work around the need for such an assumption, we have instead used the assumption that $D_{\mathcal{R}}$ is open with respect to $t$ and $Z$. In the following section, we propose an algorithm that satisfies this new requirement for constraint sets $G$ defined by arbitrary nonlinear factorable functions under very mild assumptions. Thus, this technical distinction between Theorem 7 here and the prior results in [64, 17, 19] is critical for developing practical numerical methods for systems with nonlinear constraints, despite the fact that the results in [64, 17, 19] do not explicitly assume linearity.

In the remainder of this section, we briefly describe how state bounds can be computed based on Theorem 7 and the following corollary.

**Corollary 1** *Suppose that* $\mathbf{x}^L, \mathbf{x}^U \in \mathcal{AC}(I, \mathbb{R}^{n_x})$ *are solutions of the following system of ODEs with* $i \in \{1, \ldots, n_x\}$ *and* $X(t) \equiv [\mathbf{x}^L(t), \mathbf{x}^U(t)]$*:*

$$\dot{x}_i^L(t) = \min\{\sigma_i : \boldsymbol{\sigma} \in \mathcal{R}[t, U, \mathcal{B}_i^L(X(t))]\}, \tag{3.9}$$

$$\dot{x}_i^U(t) = \max\{\sigma_i : \boldsymbol{\sigma} \in \mathcal{R}[t, U, \mathcal{B}_i^U(X(t))]\}, \tag{3.10}$$

$$x_i^L(t_0) = \min\{x_{0,i} : \mathbf{x}_0 \in X_0\}, \tag{3.11}$$

$$x_i^U(t_0) = \max\{x_{0,i} : \mathbf{x}_0 \in X_0\}. \tag{3.12}$$

*Then every solution* $(\mathbf{x}_0, \mathbf{u}, \mathbf{x}) \in X_0 \times \mathcal{U} \times \mathcal{AC}(I, \mathbb{R}^{n_x})$ *of* (4.1) *satisfies* $\mathbf{x}(t) \in X(t) \equiv [\mathbf{x}^L(t), \mathbf{x}^U(t)]$, $\forall t \in I$.

**Proof** It suffices to show that $\mathbf{x}^L$ and $\mathbf{x}^U$ satisfy the hypotheses of Theorem 7. If $(\mathbf{x}^L, \mathbf{x}^U)$ is a solution of (3.9)–(3.12) on all of $I$, then $X(t) = [\mathbf{x}^L(t), \mathbf{x}^U(t)]$ must

remain in the domain of the right-hand side functions in (3.9)–(3.10). It follows that we must have $(t, U, \mathcal{B}_i^{L/U}(X(t))) \in D_{\mathcal{R}}$, $\forall t \in I$, and hence Condition 1 of Theorem 7 holds. Moreover, Conditions 2 and 3 are directly implied by (3.11)–(3.12) and (3.9)–(3.10), respectively. □

According to Corollary 11, state bounds can be computed by simply solving the ODEs (3.9)–(3.12). Since $X_0$ is an interval and $\mathcal{R}$ is interval-valued, the min / max operations in (3.9)–(3.12) can be executed by simply selecting the $i^{\text{th}}$ upper or lower bound of the given interval. Then, (3.9)–(3.12) can be solved efficiently using any numerical integration algorithm. In fact, Assumption 3 ensures that the right-hand sides of these ODEs are locally Lipschitz continuous with respect to the states $\mathbf{x}^L$ and $\mathbf{x}^U$, as required by standard numerical solvers (see Theorem 5.5.12 in [57]).

## 3.5 An Interval Refinement Operator for Exploiting Nonlinear Constraints

This section presents a new algorithm defining the interval operator $\mathcal{R}$ in Assumption 3 for the case where the set $G$ is defined by a system of nonlinear equality and inequality constraints. This extends the work in [67, 64, 18, 19], where refinement algorithms are given for $G$ sets defined in terms of linear constraints.

**Assumption 4** *Assume that $G$ is defined by*

$$G \equiv \left\{ (t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) \in D_G : \ \mathbf{g}(t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) \leq \mathbf{0}, \ \mathbf{h}(t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) = \mathbf{0} \right\}, \tag{3.13}$$

*where $(\mathbf{g}, \mathbf{h}) : D_G \subset \mathbb{R} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \to \mathbb{R}^{n_g} \times \mathbb{R}^{n_h}$ are locally Lipschitz continuous on $D_G$ and continuously differentiable with respect to $(\mathbf{p}, \mathbf{z}, \boldsymbol{\sigma})$ at each point in $D_G$.*

Our algorithm for $\mathcal{R}$ requires inclusion functions for the derivatives of $\mathbf{g}$ and $\mathbf{h}$, as well as for the vector field $\mathbf{f} : D_f \to \mathbb{R}^{n_x}$ in (4.1a), which we formalize in the assumptions below. We use the shorthand $\mathbf{y} = (\mathbf{p}, \mathbf{z}, \boldsymbol{\sigma})$ with $n_y = n_u + n_x + n_x$ in order to write the Jacobian matrices of $\mathbf{g}$ and $\mathbf{h}$ with respect to $(\mathbf{p}, \mathbf{z}, \boldsymbol{\sigma})$ concisely as $(\frac{\partial \mathbf{g}}{\partial \mathbf{y}}, \frac{\partial \mathbf{h}}{\partial \mathbf{y}}) : D_G \to \mathbb{R}^{n_g \times n_y} \times \mathbb{R}^{n_h \times n_y}$. Additionally, for a set $D \subset \mathbb{R}^{n_x}$, we use $\mathbb{I}D$ to denote the set in $\mathbb{IR}^{n_x}$ containing all interval subsets of $D$.

**Assumption 5** *Let $D_{[G]} \subset \mathbb{I}D_G$ and let $[\frac{\partial \mathbf{g}}{\partial \mathbf{y}}] : D_{[G]} \to \mathbb{IR}^{n_g \times n_y}$ and $[\frac{\partial \mathbf{h}}{\partial \mathbf{y}}] : D_{[G]} \to \mathbb{IR}^{n_h \times n_y}$ satisfy*

1. *For every $(T, P, Z, \Sigma) \in D_{[G]}$,*

$$\left[\frac{\partial \mathbf{g}}{\partial \mathbf{y}}\right](T, P, Z, \Sigma) \supset \left\{\frac{\partial \mathbf{g}}{\partial \mathbf{y}}(t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) : (t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) \in T \times P \times Z \times \Sigma\right\}, \quad (3.14)$$

$$\left[\frac{\partial \mathbf{h}}{\partial \mathbf{y}}\right](T, P, Z, \Sigma) \supset \left\{\frac{\partial \mathbf{h}}{\partial \mathbf{y}}(t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) : (t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) \in T \times P \times Z \times \Sigma\right\}. \quad (3.15)$$

2. *$D_{[G]}$ is open.*

3. *$[\frac{\partial \mathbf{g}}{\partial \mathbf{y}}]$ and $[\frac{\partial \mathbf{h}}{\partial \mathbf{y}}]$ are locally Lipschitz continuous.*

4. *If $(T, P, Z, \Sigma) \in D_{[G]}$, then $\mathbb{I}T \times \mathbb{I}P \times \mathbb{I}Z \times \mathbb{I}\Sigma \subset D_{[G]}$.*

**Assumption 6** *Let $F : D_F \subset \mathbb{I}D_f \to \mathbb{IR}^{n_x}$ satisfy:*

1. *For every $(T, P, Z) \in D_F$,*

$$F(T, P, Z) \supset \{\mathbf{f}(t, \mathbf{p}, \mathbf{z}) : (t, \mathbf{p}, \mathbf{z}) \in I \times P \times Z\}. \quad (3.16)$$

*2. $D_F$ is open.*

*3. $F$ is locally Lipschitz continuous.*

*4. If $(T, P, Z) \in D_F$, then $\mathbb{I}T \times \mathbb{I}P \times \mathbb{I}Z \subset D_F$.*

Assumptions 5-7 are easily satisfied for a very general class of nonlinear functions. In particular, if $\frac{\partial \mathbf{g}}{\partial \mathbf{y}}$, $\frac{\partial \mathbf{h}}{\partial \mathbf{y}}$, and $\mathbf{f}$ are factorable functions (i.e., they can be written explicitly in computer code using a standard mathematics library), then $[\frac{\partial \mathbf{g}}{\partial \mathbf{y}}]$, $[\frac{\partial \mathbf{h}}{\partial \mathbf{y}}]$, and $F$ can be chosen as the natural interval extensions of $\frac{\partial \mathbf{g}}{\partial \mathbf{y}}$, $\frac{\partial \mathbf{h}}{\partial \mathbf{y}}$, and $\mathbf{f}$, respectively [48]. This satisfies Condition 1 of Assumptions 5-7 by definition. Regarding Condition 2, note that the natural interval extension of a function, say $\mathbf{f} : D_f \to \mathbb{R}^{n_x}$, is not necessarily defined for every interval subset of $D_f$ because overly conservative interval bounds on intermediate variables may cause domain violations that do not occur in real arithmetic (e.g., division by an interval containing zero). However, the maximal domain of definition $D_F \subset \mathbb{I}D_f$ is guaranteed to be open provided that each primitive univariate function appearing in the definition of $\mathbf{f}$ (e.g., $-x$, $e^x$, $x^n$, $\sin x$, etc.) is defined on an open domain, is continuous, and has a continuous interval extension there (see Corollary 2.5.35 and Remark 2.5.36 in [57]). Moreover, if these primitive univariate functions are locally Lipschitz continuous and have locally Lipschitz interval extensions, then $F$ is locally Lipschitz continuous as well (see Corollary 2.5.31 in [57]). These properties are verified for a comprehensive library of univariate functions, along with their standard interval extensions, in §2.8 of [57]. Thus, Conditions 2 and 3 of Assumptions 5-7 hold very generally. Finally, Condition 4 of Assumptions 5-7 follows directly from inclusion monotonicity of interval arithmetic (see Lemma 2.3.12 in [57]).

Recall from Assumption 3 that $\mathcal{R}$ must satisfy the following inclusion for every

$(t, P, Z)$ in its domain:

$$\mathcal{R}(t, P, Z) \supset \{\boldsymbol{\sigma} \in \mathbb{R}^{n_x} : \boldsymbol{\sigma} = \mathbf{f}(t, \mathbf{p}, \mathbf{z}),\ (t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) \in G,\ (\mathbf{p}, \mathbf{z}) \in P \times Z\}, \tag{3.17}$$

$$\supset \{\boldsymbol{\sigma} \in \mathbb{R}^{n_x} : \boldsymbol{\sigma} = \mathbf{f}(t, \mathbf{p}, \mathbf{z}),\ \mathbf{g}(t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) \leq \mathbf{0},\ \mathbf{h}(t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) = \mathbf{0},\ (\mathbf{p}, \mathbf{z}) \in P \times Z\}. \tag{3.18}$$

Our algorithm accomplishes this in three steps. First, the inclusion function $F$ is used to compute

$$\Sigma \equiv F([t, t], P, Z), \tag{3.19}$$

$$\supset \{\boldsymbol{\sigma} \in \mathbb{R}^{n_x} : \boldsymbol{\sigma} = \mathbf{f}(t, \mathbf{p}, \mathbf{z}),\ (\mathbf{p}, \mathbf{z}) \in P \times Z\}, \tag{3.20}$$

where $[t, t]$ denotes the degenerate interval containing only $t$ and the incusion follows from Condition 1 of Assumption 7. Second, the interval $P \times Z \times \Sigma$ is refined based on the constraint set $G$ to obtain a new interval

$$\hat{P} \times \hat{Z} \times \hat{\Sigma} \supset \{(\mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) \in P \times Z \times \Sigma : (t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) \in G\}. \tag{3.21}$$

Finally, the refined interval $\hat{P} \times \hat{Z} \times \hat{\Sigma}$ is used to obtain the final enclosure of $\boldsymbol{\sigma}$ (i.e., the output of $\mathcal{R}$) via

$$\mathcal{R}(t, P, Z) = \hat{\Sigma} \cap F([t, t], \hat{P}, \hat{Z}). \tag{3.22}$$

To describe the refinement step concisely, let $\mathbf{y} = (\mathbf{p}, \mathbf{z}, \boldsymbol{\sigma})$ and $Y = P \times Z \times \Sigma$. Then, the aim of the refinement step is to compute an interval enclosure of the set

$$\{\mathbf{y} \in Y : \mathbf{g}(t, \mathbf{y}) \leq \mathbf{0},\ \mathbf{h}(t, \mathbf{y}) = \mathbf{0}\}. \tag{3.23}$$

This problem has been extensively studied in the literature, e.g., see [48]. However, a unique requirement of the DI theory here is that $\mathcal{R}$ must be locally Lipschitz continuous. This implies that the refinement mapping $P \times Z \times \Sigma \mapsto \hat{P} \times \hat{Z} \times \hat{\Sigma}$ must be locally Lipschitz continuous, which is not true of most standard approaches. In this work, we propose a modified form of the interval Krawczyk method [48] that has the desired Lipschitz property.

To begin, consider the inequality constraint $g_i(t, \mathbf{y}) \leq 0$ with $i \in \{1, \ldots, n_g\}$, choose any $\tilde{\mathbf{y}} \in Y$, and let $\mathbf{y} \in Y$ satisfy $g_i(t, \mathbf{y}) \leq 0$. By the Mean Value Theorem, there must exist $\boldsymbol{\xi} \in Y$ satisfying

$$g_i(t, \mathbf{y}) = g_i(t, \tilde{\mathbf{y}}) + \frac{\partial g_i}{\partial \mathbf{y}}(t, \boldsymbol{\xi})(\mathbf{y} - \tilde{\mathbf{y}}) \leq 0. \tag{3.24}$$

Thus, there must exist $v \in V \equiv (-\infty, 0]$ satisfying

$$g_i(t, \tilde{\mathbf{y}}) + \frac{\partial g_i}{\partial \mathbf{y}}(t, \boldsymbol{\xi})(\mathbf{y} - \tilde{\mathbf{y}}) + v = 0. \tag{3.25}$$

In principle, (3.25) can be rearranged to isolate each variable $y_j$, as in

$$y_j = \tilde{y}_j - \left( \frac{\partial g_i}{\partial y_j}(t, \boldsymbol{\xi}) \right)^{-1} \left[ g_i(t, \tilde{\mathbf{y}}) + \sum_{k \neq j} \frac{\partial g_i}{\partial y_k}(t, \boldsymbol{\xi})(y_k - \tilde{y}_k) + v \right]. \tag{3.26}$$

Then, $Y_j$ can potentially be updated by evaluating the right-hand side in interval arithmetic over $(\mathbf{y}, \boldsymbol{\xi}, v) \in Y \times Y \times V$, as in the interval Hansen-Sengupta method [48]. However, this often requires division by intervals containing zero, which would violate the required Lipschitz property. Instead, we consider a more conservative approach obtained by multiplying (3.25) by an arbitrary scaling constant $\mu \in \mathbb{R}$ and

then adding $y_j - \tilde{y}_j$ on both sides, which gives

$$y_j - \tilde{y}_j = \mu \left[ g_i(t, \tilde{\mathbf{y}}) + \frac{\partial g_i}{\partial \mathbf{y}}(t, \boldsymbol{\xi})(\mathbf{y} - \tilde{\mathbf{y}}) + v \right] + y_j - \tilde{y}_j. \tag{3.27}$$

Collecting $y_j$ terms on the right, we obtain

$$y_j = \tilde{y}_j + \mu \left[ g_i(t, \tilde{\mathbf{y}}) + \sum_{k \neq j} \frac{\partial g_i}{\partial y_k}(t, \boldsymbol{\xi})(y_k - \tilde{y}_k) + v \right] + \left( 1 + \mu \frac{\partial g_i}{\partial y_j}(t, \boldsymbol{\xi}) \right) (y_j - \tilde{y}_j). \tag{3.28}$$

Since $\boldsymbol{\xi} \in Y$, it follows that $\frac{\partial g_i}{\partial \mathbf{y}}(t, \boldsymbol{\xi}) \in [\frac{\partial g_i}{\partial \mathbf{y}}](t, Y)$. Thus, (3.28) implies that the initial interval $Y_j$ can potentially be refined by the inclusion

$$y_j \in \tilde{y}_j + \mu \left[ g_i(t, \tilde{\mathbf{y}}) + \sum_{k \neq j} \left[ \frac{\partial g_i}{\partial y_k} \right] ([t,t], Y)(Y_k - \tilde{y}_k) + V \right] + \tag{3.29}$$

$$\left( 1 + \mu \left[ \frac{\partial g_i}{\partial y_j} \right] ([t,t], Y) \right) (Y_j - \tilde{y}_j). \tag{3.30}$$

Note that, in order for (3.29) to improve the original bound $Y_j$, it is desirable for the interval $(1 + \mu[\frac{\partial g_i}{\partial y_j}]([t,t], Y))$ to be a strict subset of $[-1, 1]$. In an attempt to achieve this, we apply (3.29) with two choices of $\mu$, denoted by $\mu^+$ and $\mu^-$ and defined as follows, where $\text{mid}(C)$ denotes the midpoint of an interval $C$ and $\epsilon > 0$ is a user-specified tolerance:

$$\mu^{\pm} = \pm 1 / \max \left( \epsilon, \left| \text{mid} \left( \left[ \frac{\partial g_i}{\partial y_j} \right] ([t,t], Y) \right) \right| \right). \tag{3.31}$$

These choices are motivated by the fact that, if $[\frac{\partial g_i}{\partial y_j}]([t,t], Y)$ is a singleton with magnitude greater than $\epsilon$, then either $(1 + \mu^- [\frac{\partial g_i}{\partial y_j}]([t,t], Y)) = 0$ or $(1 + \mu^+ [\frac{\partial g_i}{\partial y_j}]([t,t], Y)) = 0$. The tolerance $\epsilon$ is necessary to avoid division by zero. The complete refinement algo-

rithm for $Y$ consists of first applying the inclusion (3.29) for each choice of $g_i$ and $y_j$, with both $\mu^+$ and $\mu^-$. Next, an analogous sequence of refinements is applied using the equality constraints $\mathbf{h}(t, \mathbf{y}) = \mathbf{0}$. For these constraints, we use inclusions that are identical to (3.29) except that there is no slack variable $v$ (i.e., $V = [0, 0]$).

Algorithm 3 describes the complete procedure for evaluating $\mathcal{R}$. In lines 10, 20, and 25, $\bar{\cap}$ denotes the extended intersection $Y \bar{\cap} Z \equiv [\text{mid}(z^L, y^L, y^U), \text{mid}(z^U, y^L, y^U)]$, where $\text{mid}(a, b, c)$ denotes the middle value of $a, b, c \in \mathbb{R}$. Note that $Y \bar{\cap} Z$ agrees with $Y \cap Z$ whenever $Y \cap Z$ is nonempty, and is a singleton contained in $Y$ otherwise. This operation is used here so that $\mathcal{R}$ never returns the empty set.

**Theorem 8** *Algorithm 3 is well-defined for every input $(t, P, Z)$ in the set*

$$D_{\mathcal{R}} \equiv \{(t, P, Z) \in \mathbb{R} \times \mathbb{IR}^{n_u} \times \mathbb{IR}^{n_x} : ([t], P, Z) \in D_F,\ ([t], P, Z, F([t, t], P, Z)) \in D_{[G]}\}, \tag{3.32}$$

*where $[t]$ denotes the degenerate interval $[t, t]$. Moreover, if $\mathcal{R}(t, P, Z)$ is defined for every $(t, P, Z) \in D_{\mathcal{R}}$ by Algorithm 3, with any $\epsilon > 0$, then Assumption 3 holds.*

**Proof** Algorithm 3 is well-defined for a given $(t, P, Z) \in \mathbb{R} \times \mathbb{IR}^{n_u} \times \mathbb{IR}^{n_x}$ if and only if the arguments of the inclusion functions $F$, $[\frac{\partial \mathbf{g}}{\partial \mathbf{y}}]$, and $[\frac{\partial \mathbf{h}}{\partial \mathbf{y}}]$ in lines 2, 6, 8, 9, 11, 16, 18, 19, 21, and 25, all lie in the appropriate domain $D_F$ or $D_{[G]}$. By definition, if $(t, P, Z) \in D_{\mathcal{R}}$, then $(t, P, Z) \in D_F$ and $(t, Y) = (t, P, Z, \Sigma) \in D_{[G]}$ with $\Sigma$ defined as in line 2. By lines 10, 20, and 24, it is guaranteed that $[\frac{\partial \mathbf{g}}{\partial \mathbf{y}}]$ and $[\frac{\partial \mathbf{h}}{\partial \mathbf{y}}]$ are only ever evaluated with arguments of the form $([t, t], Y')$ with $Y' \subset Y$, and that $F$ is only ever evaluated with arguments of the form $([t, t], P', Z')$ with $P' \subset P$ and $Z' \subset Z$. But by Condition 4 of Assumptions 5–7, such arguments always lie in $D_{[G]}$ and $D_F$, respectively. Therefore, $\mathcal{R}$ is well-defined for all $(t, P, Z) \in D_{\mathcal{R}}$.

**Algorithm 3**

1: **function** $\mathcal{R}(t, P, Z)$
2: $\quad\Sigma \leftarrow F([t,t], P, Z)$
3: $\quad Y \leftarrow (P, Z, \Sigma)$
4: $\quad$**for** $i \leftarrow 1, n_g$ **do**
5: $\quad\quad$**for** $j \leftarrow 1, n_y$ **do**
6: $\quad\quad\quad\mu \leftarrow \mu^+$ (see Eq. (3.31))
7: $\quad\quad\quad\tilde{\mathbf{y}} \leftarrow \text{mid}(Y)$
8: $\quad\quad\quad\alpha \leftarrow g_i(t, \tilde{\mathbf{y}}) + \sum_{k \neq j}[\frac{\partial g_i}{\partial y_k}]([t,t], Y)(Y_k - \tilde{y}_k) + V$
9: $\quad\quad\quad\hat{Y}_j \leftarrow \tilde{y}_j + \mu\alpha + (1 + \mu[\frac{\partial g_i}{\partial y_j}]([t,t], Y))(Y_j - \tilde{y}_j)$
10: $\quad\quad\quad Y_j \leftarrow Y_j \bar{\cap} \hat{Y}_j$
11: $\quad\quad\quad\mu \leftarrow \mu^-$ (see Eq. 3.31), repeat lines 7–10
12: $\quad\quad$**end for**
13: $\quad$**end for**
14: $\quad$**for** $q \leftarrow 1, n_h$ **do**
15: $\quad\quad$**for** $j \leftarrow 1, n_y$ **do**
16: $\quad\quad\quad\mu \leftarrow \mu^+$ (see Eq. (3.31))
17: $\quad\quad\quad\tilde{\mathbf{y}} \leftarrow \text{mid}(Y)$
18: $\quad\quad\quad\alpha \leftarrow h_q(\tilde{\mathbf{y}}) + \sum_{k \neq j}[\frac{\partial h_q}{\partial y_k}]([t,t], Y)(Y_k - \tilde{y}_k)$
19: $\quad\quad\quad\hat{Y}_j \leftarrow \tilde{y}_j + \mu\alpha + (1 + \mu[\frac{\partial h_q}{\partial y_j}]([t,t], Y))(Y_j - \tilde{y}_j)$
20: $\quad\quad\quad Y_j \leftarrow Y_j \bar{\cap} \hat{Y}_j$
21: $\quad\quad\quad\mu \leftarrow \mu^-$ (see Eq. (3.31)), repeat lines 17–20
22: $\quad\quad$**end for**
23: $\quad$**end for**
24: $\quad(\hat{P}, \hat{Z}, \hat{\Sigma}) \leftarrow Y$
25: $\quad\hat{\Sigma} \leftarrow \hat{\Sigma} \bar{\cap} F([t,t], \hat{P}, \hat{Z})$
26: $\quad$**return** $\hat{\Sigma}$
27: **end function**

To verify Condition 1 of Assumption 3, choose any $(t, P, Z) \in D_{\mathcal{R}}$. By the definition of $D_{\mathcal{R}}$, we have $([t,t], P, Z) \in D_F \subset \mathbb{I}D_f$, and hence $\{t\} \times P \times Z \subset D_f$. To verify the desired inclusion, we simply argue that

$$Y \supset \{(\mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) \in \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} : \boldsymbol{\sigma} = \mathbf{f}(t, \mathbf{p}, \mathbf{z}),\ (t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) \in G,\ (\mathbf{p}, \mathbf{z}) \in P \times Z\} \tag{3.33}$$

at every stage of the algorithm. By Condition 1 of Assumption 7, this must hold immediately after line 3. Moreover, on account of (3.29), no point $\mathbf{y} = (\mathbf{p}, \mathbf{z}, \boldsymbol{\sigma})$ satisfying $\mathbf{g}(t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) \leq \mathbf{0}$ or $\mathbf{h}(t, \mathbf{p}, \mathbf{z}, \boldsymbol{\sigma}) = \mathbf{0}$ can be eliminated by the refinements in lines 8–11 or 18–21. Thus, (3.33) still holds when line 24 is reached. Applying Condition 1 of Assumption 7 again, the desired inclusion for $\mathcal{R}$ follows.

To verify Condition 3 of Assumption 3, choose any $(\hat{t}, \hat{P}, \hat{Z}) \in D_{\mathcal{R}}$. We must show that $\exists \eta > 0$ such that $(t, \hat{P}, Z) \in D_{\mathcal{R}}$ for every $t \in B_\eta(\hat{t})$ and $Z \in B_\eta(\hat{Z})$. Let $\hat{\Sigma} = F([\hat{t}, \hat{t}], \hat{P}, \hat{Z})$. By the definition of $\mathcal{D}_{\mathcal{R}}$,

$$([\hat{t}, \hat{t}], \hat{P}, \hat{Z}) \in D_F \quad \text{and} \quad ([\hat{t}, \hat{t}], \hat{P}, \hat{Z}, \hat{\Sigma}) \in D_{[G]}. \tag{3.34}$$

Since $D_{[G]}$ is open (Condition 2 of Assumption 5), $\exists \delta > 0$ such that

$$t \in B_\delta(\hat{t}),\ Z \in B_\delta(\hat{Z}),\ \Sigma \in B_\delta(\hat{\Sigma}) \quad \Longrightarrow \quad ([t,t], \hat{P}, Z, \Sigma) \in D_{[G]}. \tag{3.35}$$

Moreover, since $D_F$ is open and $F$ is continuous (Conditions 2 and 3 of Assumption

7), $\exists \eta \in (0, \delta]$ such that

$$t \in B_\eta(\hat{t}),\ Z \in B_\eta(\hat{Z}) \implies ([t,t], \hat{P}, Z) \in D_F,\ F([t,t], \hat{P}, Z) \in B_\delta(\hat{\Sigma}), \tag{3.36}$$

$$\implies ([t,t], \hat{P}, Z) \in D_F,\ ([t,t], \hat{P}, Z, F([t,t], \hat{P}, Z)) \in D_{[G]}, \tag{3.37}$$

$$\implies (t, \hat{P}, Z) \in D_{\mathcal{R}}. \tag{3.38}$$

Thus, Condition 3 of Assumption 3 holds with this choice of $\eta$.

To verify Condition 2 of Assumption 3, we argue that every line of Algorithm 3 defines its output as a locally Lipschitz continuous function of $t$ and the current value of $Y = (P, Z, \Sigma)$. Thus, Algorithm 3 defines $\mathcal{R}$ as a finite composition of locally Lipschitz functions, and it follows that $\mathcal{R}$ is locally Lipschitz continuous with respect to the input $(t, P, Z)$. By Condition 3 of Assumption 7, line 2 defines its output $\Sigma$ as a locally Lipschitz continuous function of $(t, P, Z)$, and line 3 is trivially Lipschitz continuous. Moreover, it is straightforward to show that the mapping $\mathbb{IR} \ni Q \mapsto 1/\max(\epsilon, |\text{mid}(Q)|) \in \mathbb{R}$ is Lipschitz continuous with constant $\epsilon^{-2}$. Thus, line 6 and Eq. (3.31) define $\mu$ as a Lipschitz continuous function of $[\frac{\partial g_i}{\partial y_j}]([t,t], Y)$, and hence as a locally Lipschitz continuous function of $t$ and $Y$ by Condition 3 of Assumption 5. Line 7 defines $\tilde{\mathbf{y}} = \frac{1}{2}(\mathbf{y}^L + \mathbf{y}^U)$ as Lipschitz continuous function of $Y = [\mathbf{y}^L, \mathbf{y}^U]$ with constant 1. Moreover, $g_i$ is locally Lipschitz continuous with respect to $t$ and $\tilde{\mathbf{y}}$ by Assumption 4. Thus, lines 8 and 9 define $\alpha$ and $\hat{Y}_j$ as locally Lipschitz continuous functions of $t$ and $Y$ by Condition 3 of Assumption 5 and the fact that interval addition and multiplication are locally Lipschitz continuous [48]. Finally, it is straightforward to show that the extended intersection $\bar{\cap}$ in line (10) is a Lipschitz continuous function of its arguments (see [61]). Applying analogous arguments to line 11, as well as to the equality-constraint refinements in lines 16–21, it follows that the values $(\hat{P}, \hat{Z}, \hat{\Sigma})$

in line 24 are locally Lipschitz continuous with respect to the input $(t, P, Z)$. Thus, a final application of Condition 3 of Assumption 7 to line 25 provides the required Lipschitz property of $\mathcal{R}$. $\square$

**Remark 1** *A few existing DI bounding theorems exploit state constraints of the form* $\mathbf{x}(t) \in G \subset \mathbb{R}^{n_x}$ *using a set-valued refinement operator* $\mathcal{I}_G : D_{\mathcal{I}} \rightrightarrows \mathbb{R}^{n_x}$ *satisfying* $\mathcal{I}_G(Z) \supset (Z \cap G)$ *for all* $Z \in \mathbb{IR}^{n_x}$ *[64, 18, 17, 19]. However, these results require* $\mathcal{I}_G$ *to be well-defined for every* $Z \in \mathbb{IR}^{n_x}$ *satisfying* $Z \cap G \neq \emptyset$*, which is prohibitive for nonlinear constraint sets* $G = \{\mathbf{x} : \mathbf{g}(\mathbf{x}) \leq \mathbf{0},\ \mathbf{h}(\mathbf{x}) = \mathbf{0}\}$*. Specifically, if* $\mathcal{I}_G$ *is defined using an interval Krawcyzk iteration similar to that used in Algorithm 3, then this requires* $\frac{\partial \mathbf{g}}{\partial \mathbf{x}}$ *and* $\frac{\partial \mathbf{h}}{\partial \mathbf{x}}$ *(and hence* $\mathbf{g}$ *and* $\mathbf{h}$*) to be well-defined on arbitrarily large intervals* $Z$*, which precludes the use of any constraints involving the primitive operations* $\frac{1}{x}$*,* $\ln x$*,* $\sqrt{x}$*, etc. Moreover, it is difficult to conceive of any alternative to the Krawcyzk iteration that avoids this issue. Thus, a key feature of Theorem 7 here is that there is no such assumption on* $D_{\mathcal{R}}$*. Instead, our proof of Theorem 7 uses the openness property of* $D_{\mathcal{R}}$ *asserted in Condition 3 of Assumption 3 (existing results do not assume that* $D_{\mathcal{I}}$ *is open). Theorem 8 above shows that this condition is implied by openness of* $D_{[G]}$ *and* $D_F$*, which follow from easily verifiable conditions for a very general class of nonlinear constraints, as discussed after Assumptions 5–7.*

**Remark 2** *A more efficient (but more difficult to understand) implementation of Algorithm 3 can be achieved by eliminating repeated computations that occur when computing the interval sums defining* $\alpha$ *in line 8 for every* $i$ *and* $j$*, and again in line 18 for every* $q$ *and* $j$*. Specifically, the sum computed in line 8 has* $n_y + 1$ *terms, but* $n_y - 1$ *of these are common between the sums computed for two successive values of* $j$*. An alternative approach is to evaluate the complete sum* $\alpha \leftarrow g_i(\tilde{\mathbf{y}}) + \sum_{k=1}^{n_y} [\frac{\partial g_i}{\partial y_k}]([t,t], Y)(Y_k - \tilde{y}_k) + V$ *once prior to the loop over* $j$*. Then, for*

*each $j$, this value can be corrected by simply subtracting the contribution from the interval $[\frac{\partial g_i}{\partial y_j}]([t,t],Y)(Y_j - \tilde{y}_j)$. An analogous modification can be made to line 18. This variant of Algorithm 3, which we use in all numerical experiments in §3.6, has a complexity of $O((n_g + n_h)n_y)$, as compared to $O((n_g + n_h)n_y^2)$ as written.*

## 3.6 Numerical Examples

This section demonstrates the performance of our new bounding method using two test cases. Specifically, this method consists of solving the bounding ODEs (3.9)–(3.12) with the $\mathcal{R}$ defined as in Algorithm 3 and Remark 2. Natural interval extensions were used for the inclusion functions $F$, $[\frac{\partial \mathbf{g}}{\partial \mathbf{y}}]$, and $[\frac{\partial \mathbf{h}}{\partial \mathbf{y}}]$. Numerical integration was done using the Sundials solver `CVODE` [21] with absolute and relative tolerances both as $10^{-5}$. For comparison, we also implemented the standard differential inequalities (SDI) method, which consists of solving (3.9)–(3.12) without considering the constraint set $G$, i.e., with $\mathcal{R}(t,P,Z) = F(t,P,Z)$. Both methods were implemented in `C++` on a 64-bit Linux virtual machine allocated 4GB RAM and a single core of a Dell Precision T3610 with an Intel Xeon E5-1607 v2 @ 3.00 GHz. Comparisons with published results for other state-of-the-art bounding methods are also provided.

### 3.6.1 Example 1

The two species Lotka-Volterra predator-prey model is widely used to evaluate reachable set bounding algorithms. For comparison, we use the same data as in [17, 35]. The ODEs are

$$\dot{x}(t) = u_1 x(t)(1 - y(t)), \qquad \dot{y}(t) = u_2 y(t)(x(t) - 1), \tag{3.39}$$

with horizon $I = [0, 10]$ s and initial conditions $(x, y)(t_0) = (1.2, 1.1)$. The time-invariant parameters $u_1$ and $u_2$ are uncertain with $u_1 \in [2.99, 3.01]$ and $u_2 \in [0.99, 1.01]$. The solutions of this system are known to obey one nonlinear solution invariant, regardless of the values of $u_1$ and $u_2$:

$$u_2\big[\ln(x(t)/x_0) - (x(t) - x_0)\big] + u_1\big[\ln(y(t)/y_0) - (y(t) - y_0)\big] = 0. \tag{3.40}$$

Thus, this equation can be used to define the equality constraint function $\mathbf{h}$, and hence the set $G$, as in Assumption 4.

Figure 3.1 shows that exploiting the nonlinear invariant (3.40) using our new method leads to very sharp bounds on the solutions of (3.39), whereas standard DI produces bounds that diverge after only a short integration time. Our new method required 0.024s to compute these bounds, compared to 0.002s for standard DI. By comparison, the bounds from our new method are also much tighter than those obtained using the linear programming-based polyhedral bounding algorithm in [17], which required 0.050s, and very similar to the bounds obtained using the Taylor Model algorithm in [35], which required 0.59s.

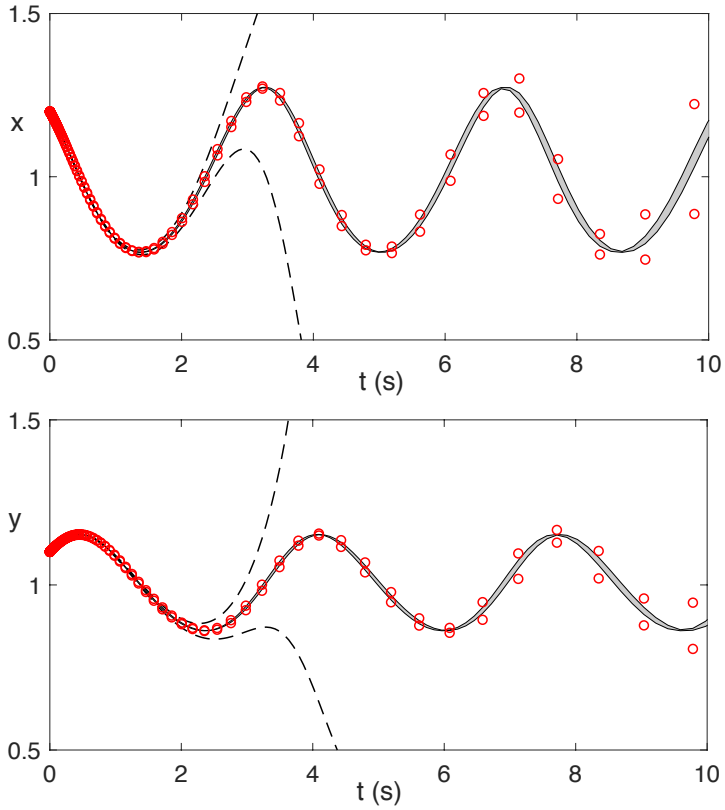


Figure 3.1: State bounds for $x$ and $y$ in (3.39) computed using SDI (dashed black) and our new method exploiting the nonlinear invariant (3.40) (red circles) with sampled solutions (gray shaded region).

### 3.6.2 Example 2

The following ODEs describe an anaerobic wastewater treatment process with pH self-regulation and liquid-gas transfer from [6]:

$$\begin{aligned}
\dot{X}_1 &= (\mu_1(S_1) - \alpha D)X_1 \\
\dot{X}_2 &= (\mu_2(S_2) - \alpha D)X_2 \\
\dot{S}_1 &= D(S_1^{in} - S_1) - k_1\mu_1(S_1)X_1 \\
\dot{S}_2 &= D(S_2^{in} - S_2) + k_2\mu_1(S_1)X_1 - k_3\mu_2(S_2)X_2 \\
\dot{Z} &= D(Z^{in} - Z) \\
\dot{C} &= D(C^{in} - C) - q_{CO_2} + k_4\mu_1(S_1)X_1 + k_5\mu_2(S_2)X_2
\end{aligned} \tag{3.41}$$

where

$$\begin{aligned}
q_{CO_2} &= k_L a(C + S_2 - Z - K_H P_{CO_2}) \\
P_{CO_2} &= \frac{\phi_{CO_2} - \sqrt{\phi_{CO_2}^2 - 4K_H P_t(C + S_2 - Z)}}{2K_H} \\
\phi_{CO_2} &= C + S_2 - Z + K_H P_t + \frac{k_6}{k_L a}\mu_2(S_2)X_2 \\
\mu_1(S_1) &= \bar{\mu}_1 \frac{S_1}{S_1 + K_{S_1}} \\
\mu_2(S_2) &= \bar{\mu}_2 \frac{S_2}{S_2 + K_{S_2} + S_2^2/K_{I_2}}
\end{aligned} \tag{3.42}$$

The time horizon is $I = [0, 20]$ days, the uncertainties are the initial conditions $X_1(t_0) \in [0.49, 0.51]$ g(COD)L$^{-1}$, $X_2(t_0) \in [0.98, 1.02]$ mmolL$^{-1}$, $C(t_0) \in [39.2, 40.8]$ mmolL$^{-1}$, and the parameters $k_1 \in [42.14, 42.98]$ g(COD) g(cell)$^{-1}$, $k_2 \in [116.5, 118.24]$ mmol g(cell)$^{-1}$. The remaining initial conditions are $S_1(t_0) = 1$ mmolL$^{-1}$, $S_2(t_0) = 5$ mmolL$^{-1}$, and $Z(t_0) = 50$ mmolL$^{-1}$, and all other parameters are constant at the

values in [77].

To the best of our knowledge, (4.39) does not obey any existing solution invariants. Thus, we apply the method in [67] to embed the model into a higher-dimensional 'lifted' system that satisfies solution invariants by design. Specifically, we define the redundant state variables

$$N_1 \equiv k_1 X_1 + S_1, \quad N_2 \equiv -k_2 X_1 + k_3 X_2 + S_2, \tag{3.43}$$

and augment (4.39) with the corresponding ODEs for $N_1$ and $N_2$ derived by differentiating (3.43). After some simplification, these are

$$\begin{aligned} \dot{N}_1 &= D(S_1^{in} + S_1(\alpha - 1) - \alpha N_1), \\ \dot{N}_2 &= D(S_2^{in} + S_2(\alpha - 1) - \alpha N_2). \end{aligned} \tag{3.44}$$

The variables $N_1$ and $N_2$ are chosen is this way because some highly nonlinear and uncertain terms cancel when deriving (3.44) from (4.39), which increases the likelihood that tighter bounds will be achieved (see [67] for a detailed discussion of this technique). By construction, the solutions of the lifted system consisting of (4.39) and (3.44) satisfy the nonlinear invariants

$$\begin{aligned} 0 &= -N_1 + k_1 X_1 + S_1, \\ 0 &= -N_2 - k_2 X_1 + k_3 X_2 + S_2. \end{aligned} \tag{3.45}$$

Thus, these equations can be used to define the equality constraint function $\mathbf{h}$, and hence the set $G$, as in Assumption 4.

Figure 3.2 compares the standard DI method applied directly to (4.39) with

our new method applied to the lifted system consisting of (4.39) and (3.44) with the nonlinear constraints (3.45). Again, standard DI produces rapidly diverging bounds. However, the use of the constraints (3.45) results in very sharp bounds over the entire time horizon, and appears to stabilize the bounds as $t \to \infty$. Our new method required $5.0 \times 10^{-2}$s to produce the bounds shown in Figure 3.2, compared to $7.0 \times 10^{-3}$s for standard DI. For reference, integrating single trajectory of (4.39) required $2.8 \times 10^{-4}$s on average. This problem was also considered in [77] over the shorter horizon $I = [0, 4]$ days, and with $k_1$ and $k_2$ fixed rather than uncertain. There, the fastest method that did not produce divergent bounds used $4^{\text{th}}$-order Taylor Models with ellipsoidal remainder bounds and required 0.41s. Thus, the use of nonlinear solution invariants provides sharp bounds at significantly lower cost in this case.

## 3.7 Appendix

The proof of Theorem 7 is based on a general result from [57] that provides sufficient conditions for two functions $\mathbf{v}, \mathbf{w} \in \mathcal{AC}(I, \mathbb{R}^{n_x})$ to bound an arbitrary function $\boldsymbol{\phi} \in \mathcal{AC}(I, \mathbb{R}^{n_x})$ (i.e., $\boldsymbol{\phi}$ need not be associated with a system of ODEs). This result is stated abstractly in terms of interval-valued mappings of the form $\Pi_i^L, \Pi_i^U : D_\Pi \subset I \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \to \mathbb{IR}$. Roughly speaking, these functions serve as generic notation for any operation that takes an interval $[\mathbf{v}, \mathbf{w}]$ as input, isolates its $i^{\text{th}}$ lower or upper face, and then refines it based on some known constraints. The specific conditions we will require of these functions are given in Hypothesis 1.

**Hypothesis 1** *For every $i \in \{1, \ldots, n_x\}$, let $\Pi_i^L, \Pi_i^U : D_\Pi \subset I \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \to \mathbb{IR}$ satisfy the following conditions:*

1. *If $(t, \mathbf{v}, \mathbf{w}) \in D_\Pi$ satisfies $\mathbf{v} \leq \boldsymbol{\phi}(t) \leq \mathbf{w}$ and $\phi_i(t) = v_i$ for some $i \in \{1, \ldots, n_x\}$, then $\dot{\phi}_i(t) \in \Pi_i^L(t, \mathbf{v}, \mathbf{w})$.*

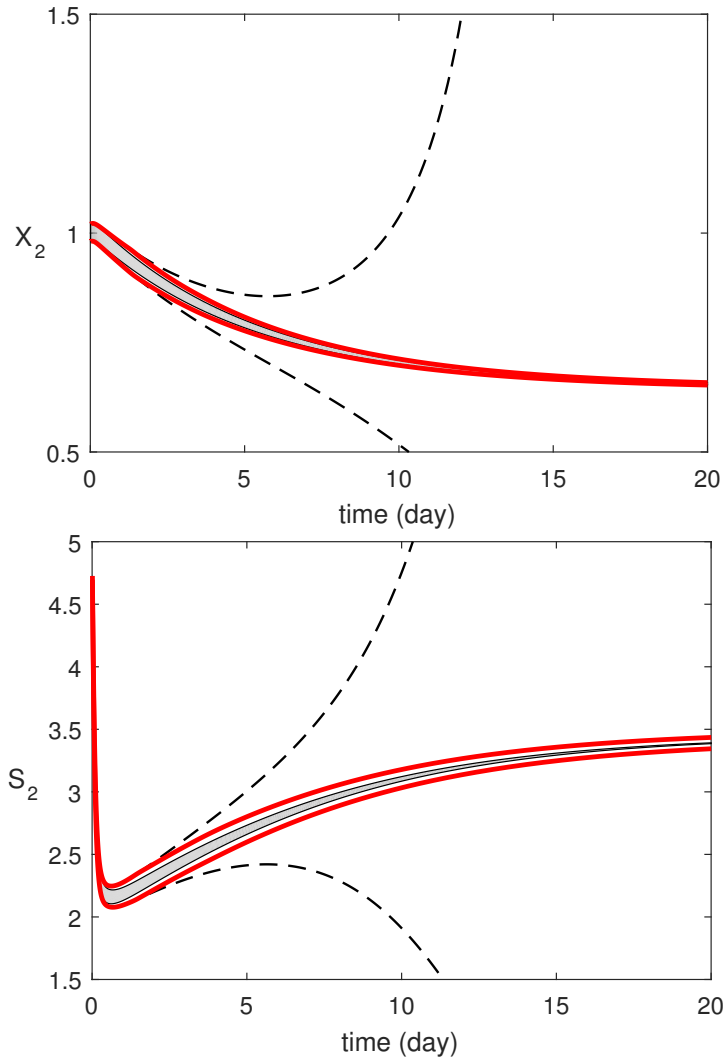


Figure 3.2: State bounds for $X_2$ and $S_2$ in (4.39) computed using SDI (dashed black) and our new method exploiting the nonlinear invariants (3.45) (solid red) with sampled solutions (gray shaded region).

2. *If* $(t, \mathbf{v}, \mathbf{w}) \in D_\Pi$ *satisfies* $\mathbf{v} \leq \boldsymbol{\phi}(t) \leq \mathbf{w}$ *and* $\phi_i(t) = w_i$ *for some* $i \in \{1, \ldots, n_x\}$, *then* $\dot{\phi}_i(t) \in \Pi_i^U(t, \mathbf{v}, \mathbf{w})$.

3. $D_\Pi$ *is open with respect to the set* $A \equiv \{(t, \mathbf{v}, \mathbf{w}) \in I \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} : \mathbf{v} \leq \mathbf{w}\}$. *Specifically, for any* $(\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}}) \in D_\Pi \cap A$, *there exists* $\eta > 0$ *such that* $B_\eta((\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}})) \cap A$ *is a subset of* $D_\Pi$.

4. $\Pi_i^L$ *and* $\Pi_i^U$ *are locally Lipschitz continuous with respect to* $\mathbf{v}$ *and* $\mathbf{w}$, *uniformly with respect to* $t$. *Specifically, for any* $(\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}}) \in D_\Pi$, *there exists* $\eta > 0$ *and* $\alpha \in L^1(I)$ *such that*

$$d_H(\Pi_i^L(t, \mathbf{v}, \mathbf{w}), \Pi_i^L(t, \tilde{\mathbf{v}}, \tilde{\mathbf{w}})) \leq \alpha(t) \max\left(\|\mathbf{v} - \tilde{\mathbf{v}}\|, \|\mathbf{w} - \tilde{\mathbf{w}}\|\right), \tag{3.46}$$

$$d_H(\Pi_i^U(t, \mathbf{v}, \mathbf{w}), \Pi_i^U(t, \tilde{\mathbf{v}}, \tilde{\mathbf{w}})) \leq \alpha(t) \max\left(\|\mathbf{v} - \tilde{\mathbf{v}}\|, \|\mathbf{w} - \tilde{\mathbf{w}}\|\right), \tag{3.47}$$

*for every* $(t, \mathbf{v}, \mathbf{w}), (t, \tilde{\mathbf{v}}, \tilde{\mathbf{w}}) \in B_\eta((\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}})) \cap D_\Pi$.

Theorem 15 is the central result we will use to prove Theorem 7. It is proven as Theorem 3.5.1 in [57] under slightly different hypotheses, as discussed below.

**Theorem 9** *Let* $\boldsymbol{\phi}, \mathbf{v}, \mathbf{w} \in \mathcal{AC}(I, \mathbb{R}^{n_x})$ *satisfy*

1. $(t, \mathbf{v}(t), \mathbf{w}(t)) \in D_\Pi, \forall t \in I$.

2. $\mathbf{v}(t_0) \leq \boldsymbol{\phi}(t_0) \leq \mathbf{w}(t_0)$.

3. *For a.e.* $t \in I$ *and each index* $i$,

   (a) $\dot{v}_i(t) \leq \sigma_i$ *for all* $\sigma_i \in \Pi_i^L(t, \mathbf{v}(t), \mathbf{w}(t))$,

   (b) $\dot{w}_i(t) \geq \sigma_i$ *for all* $\sigma_i \in \Pi_i^U(t, \mathbf{v}(t), \mathbf{w}(t))$.

*If Hypothesis 1 holds, then* $\mathbf{v}(t) \leq \boldsymbol{\phi}(t) \leq \mathbf{w}(t), \forall t \in I$.

In [57], Theorem 15 is proven with a modified version of Hypothesis 1, which is stated explicitly as Hypothesis 2 below. We prefer Hypothesis 1 here because it is easier to verify when using Theorem 15 to prove Theorem 7. Moreover, the conditions of Hypothesis 1 are much easier to understand, whereas Hypothesis 2 is very abstract. In Lemma 1, we show that Hypothesis 1 implies Hypothesis 2, so that Theorem 15 follows immediately from Theorem 3.5.1 in [57].

**Hypothesis 2** *For every* $i \in \{1, \ldots, n_x\}$, *let* $\Pi_i^L, \Pi_i^U : D_\Pi \subset I \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \rightrightarrows \mathbb{R}$ *(i.e.,* $\Pi_i^L(t, \mathbf{v}, \mathbf{w})$ *and* $\Pi_i^U(t, \mathbf{v}, \mathbf{w})$ *are subsets of* $\mathbb{R}$*, not necessarily intervals). Assume that, given any* $(\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}}) \in D_\Pi$ *satisfying* $\hat{\mathbf{v}} \leq \boldsymbol{\phi}(\hat{t}) \leq \hat{\mathbf{w}}$ *and either* $\phi_i(\hat{t}) = \hat{v}_i$ *or* $\phi_i(\hat{t}) = \hat{w}_i$ *for at least one* $i \in \{1, \ldots, n_x\}$*, there exists* $\eta > 0$ *and* $\alpha \in L^1(I)$ *such that the following conditions hold for every* $(t, \mathbf{v}, \mathbf{w}) \in B_\eta((\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}})) \cap D_\Pi$*:*

1. *If* $\phi_i(t) < v_i$*, then* $\exists \sigma_i \in \Pi_i^L(t, \mathbf{v}, \mathbf{w})$ *such that*

$$|\sigma_i - \dot{\phi}_i(t)| \leq \alpha(t) \max(\| \max(\mathbf{v} - \boldsymbol{\phi}(t), \mathbf{0})\|, \| \max(\boldsymbol{\phi}(t) - \mathbf{w}, \mathbf{0})\|). \tag{3.48}$$

2. *If* $\phi_i(t) > w_i$*, then* $\exists \sigma_i \in \Pi_i^U(t, \mathbf{v}, \mathbf{w})$ *such that* (3.48) *holds.*

**Lemma 1** *If* $\Pi_i^L$ *and* $\Pi_i^U$ *satisfy Hypothesis 1, then they also satisfy Hypothesis 2.*

**Proof** Assume that Hypothesis 1 holds. To verify Hypothesis 2, choose any $(\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}}) \in D_\Pi$ such that $\hat{\mathbf{v}} \leq \boldsymbol{\phi}(\hat{t}) \leq \hat{\mathbf{w}}$ and either $\phi_i(\hat{t}) = \hat{v}_i$ or $\phi_i(\hat{t}) = \hat{w}_i$ for at least one $i \in \{1, \ldots, n_x\}$. Define

$$\underline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w}) \equiv \min(\mathbf{v}, \boldsymbol{\phi}(t)) \text{ and } \overline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w}) \equiv \max(\mathbf{w}, \boldsymbol{\phi}(t)), \ \forall (t, \mathbf{v}, \mathbf{w}) \in I \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_x}. \tag{3.49}$$

Note that $(\underline{\boldsymbol{\phi}}(\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}}), \overline{\boldsymbol{\phi}}(\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}})) = (\hat{\mathbf{v}}, \hat{\mathbf{w}})$ and $\underline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w}) \leq \boldsymbol{\phi}(t) \leq \overline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w})$, $\forall (t, \mathbf{v}, \mathbf{w}) \in I \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_x}$.

With $(\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}})$ as above, let $\eta_{C3} > 0$ satisfy Condition 3 of Hypothesis 1, and let $\eta_{C4} > 0$ and $\alpha \in L^1(I)$ satisfy Condition 4 of Hypothesis 1. Set $\eta_C = \min(\eta_{C3}, \eta_{C4})$. Since $(\underline{\boldsymbol{\phi}}(\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}}), \overline{\boldsymbol{\phi}}(\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}})) = (\hat{\mathbf{v}}, \hat{\mathbf{w}})$ and the functions $\underline{\boldsymbol{\phi}}$ and $\overline{\boldsymbol{\phi}}$ are continuous, we may choose $\eta \in (0, \eta_C]$ such that

$$(t, \mathbf{v}, \mathbf{w}) \in B_\eta((\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}})) \quad \Longrightarrow \quad (t, \underline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w}), \overline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w})) \in B_{\eta_C}((\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}})). \quad (3.50)$$

By Condition 3 of Hypothesis 1, it follows that

$$(t, \mathbf{v}, \mathbf{w}) \in B_\eta((\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}})) \text{ and } t \in I \quad \Longrightarrow \quad (t, \underline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w}), \overline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w})) \in D_\Pi. \quad (3.51)$$

We now show that Hypothesis 2 holds with this choice of $\eta$ and $\alpha$. To verify Condition 1 of Hypothesis 2, choose any $(t, \mathbf{v}, \mathbf{w}) \in B_\eta((\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}})) \cap D_\Pi$ such that $\phi_i(t) < v_i$. We will apply the Lipschitz condition (3.46) with this choice of $\mathbf{v}$ and $\mathbf{w}$ and with $\tilde{\mathbf{v}} = \underline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w})$ and $\tilde{\mathbf{w}} = \overline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w})$. To see that this condition is applicable, first note that $(t, \mathbf{v}, \mathbf{w}) \in B_{\eta_{C4}}((\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}}))$ because $\eta \leq \eta_{C4}$. Moreover, in light of (3.50) and (3.51), we are guaranteed that $(t, \tilde{\mathbf{v}}, \tilde{\mathbf{w}}) \in B_{\eta_{C4}}((\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}})) \cap D_\Pi$. Thus, (3.46) gives

$$\begin{aligned}
&d_H(\Pi_i^L(t, \mathbf{v}, \mathbf{w}), \Pi_i^L(t, \underline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w}), \overline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w}))) && (3.52)\\
&\leq \alpha(t) \max\left(\|\mathbf{v} - \underline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w})\|, \|\mathbf{w} - \overline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w})\|\right), && (3.53)\\
&= \alpha(t) \max(\|\max(\mathbf{v} - \boldsymbol{\phi}(t), \mathbf{0})\|, \|\max(\boldsymbol{\phi}(t) - \mathbf{w}, \mathbf{0})\|).
\end{aligned}$$

Next, we apply Condition 1 of Hypothesis 1 to the point $(t, \underline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w}), \overline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w})) \in D_\Pi$. This is possible because $\underline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w}) \leq \boldsymbol{\phi}(t) \leq \overline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w})$ and $\phi_i(t) = \min(\phi_i(t), v_i) = \underline{\phi}_i(t, \mathbf{v}, \mathbf{w})$. Thus, Condition 1 of Hypothesis 1 gives $\dot{\phi}_i(t) \in \Pi_i^L(t, \underline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w}), \overline{\boldsymbol{\phi}}(t, \mathbf{v}, \mathbf{w}))$. Then, by the definition of the Hausdorff metric, (3.52) implies that $\exists \sigma_i \in \Pi_i^L(t, \mathbf{v}, \mathbf{w})$

satisfying (3.48). This proves Condition 1 of Hypothesis 2, and Condition 2 follows from an analogous argument. □

To prove Theorem 7, we will apply Theorem 15 with the following definitions:

$$D_\Pi \equiv \left\{ \begin{array}{lll} t & \in I & \mathbf{v} \leq \mathbf{w} \\ \mathbf{v} & \in \mathbb{R}^{n_x} \ : & (t, U, \mathcal{B}_i^{L/U}([\mathbf{v}, \mathbf{w}])) \in D_\mathcal{R} \\ \mathbf{w} & \in \mathbb{R}^{n_x} & \forall i \in \{1, \ldots, n_x\} \end{array} \right\} \tag{3.54}$$

$$\Pi_i^L(t, \mathbf{v}, \mathbf{w}) \equiv \{\sigma_i \in \mathbb{R} : \boldsymbol{\sigma} \in \mathcal{R}[t, U, \mathcal{B}_i^L([\mathbf{v}, \mathbf{w}])]\} \tag{3.55}$$

$$\Pi_i^U(t, \mathbf{v}, \mathbf{w}) \equiv \{\sigma_i \in \mathbb{R} : \boldsymbol{\sigma} \in \mathcal{R}[t, U, \mathcal{B}_i^U([\mathbf{v}, \mathbf{w}])]\} \tag{3.56}$$

**Lemma 2** *Let* $(\mathbf{x}_0, \mathbf{u}, \mathbf{x}) \in X_0 \times \mathcal{U} \times \mathcal{AC}(I, \mathbb{R}^{n_x})$ *be any solution of* (4.1)*. Under Assumption 3, the definitions* (3.54)–(3.56) *satisfy Hypothesis 1 with* $\boldsymbol{\phi} \equiv \mathbf{x}$*.*

**Proof** To verify Condition 1 of Hypothesis 1, choose any $(t, \mathbf{v}, \mathbf{w}) \in D_\Pi$ such that $\mathbf{v} \leq \boldsymbol{\phi}(t) \leq \mathbf{w}$ and $\phi_i(t) = v_i$ for some $i \in \{1, \ldots, n_x\}$. These conditions imply that $\mathbf{x}(t) = \boldsymbol{\phi}(t) \in \mathcal{B}_i^L([\mathbf{v}, \mathbf{w}])$. Moreover, by the definition of $D_\Pi$, $(t, \mathbf{v}, \mathbf{w}) \in D_\Pi$ implies that $(t, U, \mathcal{B}_i^L([\mathbf{v}, \mathbf{w}])) \in D_\mathcal{R}$. Since $(\mathbf{x}_0, \mathbf{u}, \mathbf{x})$ is a solution of (4.1), Condition 1 of Assumption 3 implies that

$$\dot{\mathbf{x}}(t) \in \mathcal{R}\left[t, U, \mathcal{B}_i^L([\mathbf{v}, \mathbf{w}])\right]. \tag{3.57}$$

By (3.55), it follows that $\dot{\phi}_i(t) = \dot{x}_i(t) \in \Pi_i^L(t, \mathbf{v}, \mathbf{w})$. This proves Condition 1 of Hypothesis 1, and Condition 2 follows from an analogous argument.

To verify Condition 3 of Hypothesis 1, choose any $(\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}}) \in D_\Pi \cap A$. By (3.54), $(\hat{t}, U, \mathcal{B}_i^{L/U}([\hat{\mathbf{v}}, \hat{\mathbf{w}}])) \in D_\mathcal{R}$ for all $i \in \{1, \ldots, n_x\}$. By Condition 3 of Assumption 3,

$D_{\mathcal{R}}$ is open with respect to $t$ and $Z$. Thus, there must exist $\eta > 0$ such that

$$(t, Z) \in B_\eta(\hat{t}) \times B_\eta(\mathcal{B}_i^{L/U}([\hat{\mathbf{v}}, \hat{\mathbf{w}}])) \quad \Longrightarrow \quad (t, U, Z) \in D_{\mathcal{R}}. \tag{3.58}$$

Moreover, by the definition of $\mathcal{B}_i^{L/U}$, it follows that

$$\begin{aligned}(t, Z) \in B_\eta(\hat{t}) \times B_\eta([\hat{\mathbf{v}}, \hat{\mathbf{w}}]) \quad &\Longrightarrow \quad (t, \mathcal{B}_i^{L/U}(Z)) \in B_\eta(\hat{t}) \times B_\eta(\mathcal{B}_i^{L/U}([\hat{\mathbf{v}}, \hat{\mathbf{w}}])), \\ &\Longrightarrow \quad (t, U, \mathcal{B}_i^{L/U}(Z)) \in D_{\mathcal{R}}.\end{aligned} \tag{3.59}$$

We claim that Condition 3 of Hypothesis 1 holds with this $\eta$. To see this, choose any $(t, \mathbf{v}, \mathbf{w}) \in B_\eta((\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}})) \cap A$. It suffices to show that $(t, \mathbf{v}, \mathbf{w}) \in D_\Pi$. Since $(t, \mathbf{v}, \mathbf{w}) \in A$, we have $t \in I$ and $\mathbf{v} \leq \mathbf{w}$. Moreover, since $(t, \mathbf{v}, \mathbf{w}) \in B_\eta((\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}}))$, it follows from the definition of $d_H$ that $d_H([\mathbf{v}, \mathbf{w}], [\hat{\mathbf{v}}, \hat{\mathbf{w}}]) \leq \eta$. Finally, since $|t - \hat{t}| \leq \eta$ as well, (3.59) ensures that $(t, U, \mathcal{B}_i^{L/U}([\mathbf{v}, \mathbf{w}])) \in D_{\mathcal{R}}$. Thus, by (3.54), $(t, \mathbf{v}, \mathbf{w}) \in D_\Pi$, as desired.

To verify Condition 4 of Hypothesis 1, choose any $(\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}}) \in D_\Pi$. By (3.54), $(\hat{t}, U, \mathcal{B}_i^{L/U}([\hat{\mathbf{v}}, \hat{\mathbf{w}}])) \in D_{\mathcal{R}}$ for all $i \in \{1, \ldots, n_x\}$. Thus, by Condition 2 of Assumption 3, there exists $\eta, L > 0$ such that

$$d_H(\mathcal{R}(t, U, Z), \mathcal{R}(t, U, \tilde{Z})) \leq L d_H(Z, \tilde{Z}), \tag{3.60}$$

for every $t \in B_\eta(\hat{t})$ and $Z, \tilde{Z} \in B_\eta(\mathcal{B}_i^{L/U}([\hat{\mathbf{v}}, \hat{\mathbf{w}}]))$. We claim that Condition 4 of Hypothesis 1 holds with this choice of $\eta$ and $\alpha = L$. To see this, choose any

$(t, \mathbf{v}, \mathbf{w}), (t, \tilde{\mathbf{v}}, \tilde{\mathbf{w}}) \in B_\eta((\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}})) \cap D_\Pi$. It suffices to show that

$$d_H(\Pi_i^L(t, \mathbf{v}, \mathbf{w}), \Pi_i^L(t, \tilde{\mathbf{v}}, \tilde{\mathbf{w}})) \leq L \max\left(\|\mathbf{v} - \tilde{\mathbf{v}}\|, \|\mathbf{w} - \tilde{\mathbf{w}}\|\right), \tag{3.61}$$

$$d_H(\Pi_i^U(t, \mathbf{v}, \mathbf{w}), \Pi_i^U(t, \tilde{\mathbf{v}}, \tilde{\mathbf{w}})) \leq L \max\left(\|\mathbf{v} - \tilde{\mathbf{v}}\|, \|\mathbf{w} - \tilde{\mathbf{w}}\|\right). \tag{3.62}$$

By (3.55) and the definition of the Hausdorff metric $d_H$,

$$d_H(\Pi_i^L(t, \mathbf{v}, \mathbf{w}), \Pi_i^L(t, \tilde{\mathbf{v}}, \tilde{\mathbf{w}})) \leq d_H(\mathcal{R}(t, U, \mathcal{B}_i^L([\mathbf{v}, \mathbf{w}])), \mathcal{R}(t, U, \mathcal{B}_i^L([\tilde{\mathbf{v}}, \tilde{\mathbf{w}}]))). \tag{3.63}$$

But, as argued above, the fact that $(t, \mathbf{v}, \mathbf{w})$ and $(t, \tilde{\mathbf{v}}, \tilde{\mathbf{w}})$ are elements of $B_\eta((\hat{t}, \hat{\mathbf{v}}, \hat{\mathbf{w}}))$ implies that $\mathcal{B}_i^L([\mathbf{v}, \mathbf{w}])$ and $\mathcal{B}_i^L([\tilde{\mathbf{v}}, \tilde{\mathbf{w}}])$ are elements of $B_\eta(\mathcal{B}_i^{L/U}([\hat{\mathbf{v}}, \hat{\mathbf{w}}]))$. Then, using (3.60), we have

$$d_H(\Pi_i^L(t, \mathbf{v}, \mathbf{w}), \Pi_i^L(t, \tilde{\mathbf{v}}, \tilde{\mathbf{w}})) \leq L d_H(\mathcal{B}_i^L([\mathbf{v}, \mathbf{w}]), \mathcal{B}_i^L([\tilde{\mathbf{v}}, \tilde{\mathbf{w}}])), \tag{3.64}$$

$$\leq L \max\left(\|\mathbf{v} - \tilde{\mathbf{v}}\|, \|\mathbf{w} - \tilde{\mathbf{w}}\|\right), \tag{3.65}$$

as desired. The proof of (3.62) is analogous. □

We now prove Theorem 7. Choose any $\mathbf{x}^L, \mathbf{x}^U \in \mathcal{AC}(I, \mathbb{R}^{n_x})$ and suppose that Conditions 1–3 of Theorem 7 hold. Moreover, let $(\mathbf{x}_0, \mathbf{u}, \mathbf{x}) \in X_0 \times \mathcal{U} \times \mathcal{AC}(I, \mathbb{R}^{n_x})$ be any solution of (4.1). We show that the hypotheses of Theorem 15 are satisfied with $\mathbf{v} = \mathbf{x}^L$, $\mathbf{w} = \mathbf{x}^U$, $\boldsymbol{\phi} = \mathbf{x}$, and the definitions (3.54)–(3.56). As a consequence, $\mathbf{x}(t) \in [\mathbf{x}^L(t), \mathbf{x}^U(t)]$, $\forall t \in I$, as desired.

With the definition of $D_\Pi$ in (3.54), Condition 1 of Theorem 15 follows directly from Condition 1 of Theorem 7. Condition 2 of Theorem 15 also follows directly from Condition 2 of Theorem 7 since $\boldsymbol{\phi}(t_0) = \mathbf{x}(t_0) = \mathbf{x}_0 \in X_0 \subset [\mathbf{x}^L(t_0), \mathbf{x}^U(t_0)] = [\mathbf{v}(t_0), \mathbf{w}(t_0)]$. Finally, Condition 3 of Theorem 15 follows from Condition 3 of Theo-

rem 7. To see this, choose any $\sigma_i \in \Pi_i^L(t, \mathbf{v}(t), \mathbf{w}(t)) = \Pi_i^L(t, \mathbf{x}^L(t), \mathbf{x}^U(t))$. By (3.55), there must exist $\sigma_j$, $\forall j \neq i$, such that

$$\boldsymbol{\sigma} \in \mathcal{R}[t, U, \mathcal{B}_i^L([\mathbf{x}^L(t), \mathbf{x}^U(t)])]. \tag{3.66}$$

Therefore, by Condition 3(a) of Theorem 7, we must have $\dot{v}_i(t) \leq \sigma_i$. This proves Condition 3(a) of Theorem 15. Condition 3(b) is proven analogously. Since all of the hypotheses of Theorem 15 are met, we conclude that

$$\mathbf{x}(t) = \boldsymbol{\phi}(t) \in [\mathbf{v}(t), \mathbf{w}(t)] = [\mathbf{x}^L(t), \mathbf{x}^U(t)], \quad \forall t \in I. \tag{3.67}$$

This completes the proof of Theorem 7.

# Chapter 4

# Tight Reachability Bounds for Constrained Nonlinear Systems Using Mean Value Differential Inequalities

## 4.1 Introduction

This chapter presents a new method for rigorously bounding the set of trajectories consistent with a given system of nonlinear ordinary differential equations (ODEs) subject to bounded, time-invariant uncertainties, and consistent with a given set of constraints in the joint state-and-uncertainty space (the constraints may be trivial, so standard reachability analysis is a special case). Such reachability bounds are important in algorithms for set-based state estimation [54, 42], fault detection and diagnosis [60, 53], robust predictive control [78, 15], and the global solution of open-loop optimal control problems [62, 22]. Accordingly, a wide variety of bounding methods

have been developed [41, 64, 67, 4, 34, 24], see [64, 67] for an overview. However, these methods often provide an unworkable balance between computational cost and bound accuracy for systems with strong nonlinearities or large uncertainties, which prevents their use for online computations in many important control applications.

The method presented in this chapter is an extension of existing methods based on differential inequalities (DI). For a given system of ODEs, the standard DI method applies simple interval arithmetic to derive an auxiliary system of ODEs that describes time-varying interval reachability bounds as its solutions. This is very efficient, but often provides extremely conservative bounds. In recent years, this drawback of DI has been addressed by two broad strategies. First, the DI approach has been extended to enable the use of more complex bounding sets, including polytopes and Taylor models with interval or ellipsoidal remainder bounds [17, 19, 77]. Such methods can be highly accurate when sufficiently complex sets are used, but the computational cost is often much higher than the standard interval DI method [19, 77]. Second, interval-based DI methods have been developed that achieve tighter bounds by exploiting model redundancy [66, 69, 64, 18, 67]. In the most general approach [67], new state variables are defined as user-specified functions of the original states, leading to a higher-dimensional system whose solutions obey a set of redundant algebraic relationships by design. A DI method is then applied to this larger system wherein the redundant relationships are used to refine the computed bounds continuously as they are propagated forward in time. Examples in [67] show that this can produce much tighter bounds than standard DI. Moreover, although the cost is significantly higher than standard DI, comparisons to date suggest that this approach can be much more efficient than DI methods based on more complex sets [67, 69]. However, this approach is not automated, and choosing effective new states typically requires considerable insight.

In this chapter, we develop a third approach to reduce the conservatism of DI methods. To motivate this approach, consider the simpler problem of bounding the range of an algebraic expression $f(\mathbf{x})$ [48]. Here too, interval arithmetic is well-known to produce conservative bounds. However, tighter bounds can often be achieved by a variety of advanced methods called centered forms [48]. The simplest of these is the mean value form, which uses bounds on $\frac{\partial f}{\partial \mathbf{x}}$ to obtain a sharper enclosure of $f(\mathbf{x})$ using the Mean Value Theorem. The objective of this chapter is to extend this method to dynamic systems by combining it with the DI approach. Specifically, we augment the ODEs of interest with their forward sensitivity system (w.r.t. uncertain initial conditions and parameters) and apply a DI method to obtain time-varying bounds on both the original states and the sensitivities. By bounding the sensitivities, it becomes possible to refine the computed state bounds at any point in time using a mean value form enclosure. For this refinement, we apply an algorithm similar to those used to exploit redundant algebraic relationships in redundancy-based DI methods [69, 66]. In fact, our new mean-value DI method can be viewed as a method for automating the redundancy-based DI approach, specifically by choosing the new states as the forward sensitivities and observing that these are (approximately) algebraically related to the original states by a first-order Taylor expansion. Although this redundant relation is only approximate, we show that a valid refinement procedure is still possible through the Mean Value Theorem. Moreover, we show that mean-value DI can be seamlessly integrated with existing redundancy-based DI methods by developing a refinement procedure that combines mean value enclosures of the states with other redundant relationships, e.g., derived manually as in [67].

The results in this chapter are closely related to several other recent developments. Advanced DI methods based on Taylor models and affine arithmetic [19, 77] also make use of forward sensitivities for describing the bounding sets. However,

these methods do not use mean value enclosures specifically. Moreover, with the exception of [19], these approaches do not use the relationship between the states and sensitivities at each point in time to provide a continuous bound refinement, as we do here. The paper [50] provides advanced methods for bounding forward and adjoint sensitivities, but does not use these bounds to refine the bounds on the states.

The remainder of the chapter is organized as follows. A formal problem statement is given in §4.2. The supporting theory for our new mean-value DI approach is then presented in §4.3, and implementation is discussed in §4.5. The accuracy of the computed bounds and the related second-order convergence property is discussed in §4.6. Finally, case studies are presented in §4.7.

### 4.1.1 Preliminaries and Notation

For any $\mathbf{y}^L, \mathbf{y}^U \in \mathbb{R}^n$, let $Y = [\mathbf{y}^L, \mathbf{y}^U]$ denote the compact $n$-dimensional interval vector $\{\mathbf{y} \in \mathbb{R}^n : \mathbf{y}^L \leq \mathbf{y} \leq \mathbf{y}^U\}$. Moreover, define the midpoint $\text{mid}(Y) \equiv \frac{1}{2}(\mathbf{y}^L + \mathbf{y}^U)$, the width $w(Y) \equiv \|\mathbf{y}^U - \mathbf{y}^L\|_\infty$, and the magnitude vector $|Y| \equiv (\max(|y_1^L|, |y_1^U|), \cdots, \max(|y_n^L|, |y_n^U|))$. Let $\mathbb{IR}^n$ and $\mathbb{IR}^{n\times m}$ denote the set of all nonempty $n$-dimensional interval vectors and $n$-by-$m$ interval matrices, respectively. Similarly, for $D \subset \mathbb{R}^n$, the set of all nonempty interval subsets of $D$ is denoted by $\mathbb{I}D$. Let $D \subset \mathbb{R}^n$ and $\mathbf{f} : D \to \mathbb{R}^m$. A mapping $F : \mathcal{D} \subset \mathbb{I}D \to \mathbb{IR}^m$ is an *inclusion function* for $\mathbf{f}$ on $\mathcal{D}$ if

$$F(X) \supset \mathbf{f}(X) \equiv \{\mathbf{f}(\mathbf{x}) : \mathbf{x} \in X\}, \quad \forall X \in \mathcal{D}.$$

Inclusion functions can be readily derived for *factorable functions*, which are functions that can be evaluated by a finite recursive composition of binary additions, binary multiplications, and standard univariate functions such as $-x$, $\frac{1}{x}$, $x^n$, $e^x$, etc. This

includes nearly every function that can be written explicitly in computer code. For any factorable function $\mathbf{f}$, a specific inclusion function called the *natural interval extension* can be constructed by simply replacing each operation in the definition of $\mathbf{f}$ with a suitable interval counterpart [48]. In the following sections, we make use of several properties of natural interval extensions from [57].

The Hausdorff distance $d_H$ induced by $\|\cdot\|_\infty$ is a metric on $\mathbb{IR}^n$ [48]. Therefore, standard definitions and results concerning sets and functions on metric spaces are applicable. For example, the open ball of radius $\eta > 0$ centered at $X \in \mathbb{IR}^n$ is defined by $B_\eta(X) \equiv \{Z \in \mathbb{IR}^n : d_H(X, Z) < \eta\}$. Similarly, a set $\mathcal{X} \subset \mathbb{IR}^n$ (i.e., $\mathcal{X}$ is a set whose elements are intervals) is called open if for every $X \in \mathcal{X}$, $\exists \eta > 0$ such that $B_\eta(X) \subset \mathcal{X}$. Additionally, $F : \mathcal{D} \subset \mathbb{IR}^n \to \mathbb{IR}^m$ is called locally Lipschitz continuous on $\mathcal{D}$ if for every $X \in \mathcal{D}$, $\exists L, \eta > 0$ such that $d_H(F(\bar{X}), F(\hat{X})) \leq L d_H(\bar{X}, \hat{X})$ for every $\bar{X}, \hat{X} \in B_\eta(X) \cap \mathcal{D}$. These definitions will be used in conjunction with the standard facts that the pre-image of an open set under a continuous function is open, and that the composition of locally Lipschitz functions is locally Lipschitz, both of which hold in general metric spaces.

## 4.2 Problem Statement

Let $I = [t_0, t_f] \subset \mathbb{R}$ be a time horizon of interest and let $P \in \mathbb{IR}^{n_p}$ be a compact interval of time-invariant uncertain parameters $\mathbf{p}$. Let $\mathbf{f} : D_f \subset \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_x}$ and $\mathbf{x}_0 : P \to \mathbb{R}^{n_x}$ be continuously differentiable functions. Let $G \subset \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_p}$

be a constraint set, and consider the dynamic system described by

$$\dot{\mathbf{x}}(t, \mathbf{p}) = \mathbf{f}(t, \mathbf{x}(t, \mathbf{p}), \mathbf{p}), \tag{4.1a}$$

$$\mathbf{x}(t_0, \mathbf{p}) = \mathbf{x}_0(\mathbf{p}), \tag{4.1b}$$

$$(t, \mathbf{x}(t, \mathbf{p}), \mathbf{p}) \in G. \tag{4.1c}$$

It is assumed that there is a unique continuously differentiable solution $\mathbf{x} : I \times P \to \mathbb{R}^{n_x}$ satisfying (4.1a)-(4.1b) (but not necessarily (4.1c)) for every $(t, \mathbf{p}) \in I \times P$.

**Definition 11** *Define the feasible parameter set* $P^* \equiv \{\mathbf{p} \in P : (t, \mathbf{x}(t, \mathbf{p}), \mathbf{p}) \in G,\ \forall t \in I\}$. *The reachable set of the constrained system* (4.1) *is defined for every* $t \in I$ *as*

$$Re(t) \equiv \{\mathbf{x}(t, \mathbf{p}) : \mathbf{p} \in P^*\}. \tag{4.2}$$

We are interested in computing a tight, time-varying enclosure of $Re(t)$. In other words, we wish to bound all solutions of (4.1a)–(4.1b) that have $\mathbf{p} \in P$ and satisfy (4.1c). Note that we do not use reachability analysis to prove satisfaction of (4.1c), as in verification problems. Rather, we are interested in bounding only those trajectories that are feasible in (4.1c). This problem is of interest both when $G$ is a true constraint (i.e., it is potentially violated by some trajectories) and when $G$ is redundant with (4.1a)–(4.1b) (i.e., all solutions of (4.1a)–(4.1b) are known in advance to satisfy (4.1c)). In the former case, bounding $Re(t)$ is useful in algorithms for solving optimal control problems with path constraints to guaranteed global optimality [22]. In the later case, $P^* = P$ and $Re(t)$ reduces to the standard reachable set, which is useful to bound in a variety of applications. Formulating the problem with a constraint set $G$ is still useful in this case because redundant constraints such as conservation laws

and other solution invariants can be used to achieve much tighter reachability bounds than those obtained by considering (4.1a)–(4.1b) alone, specifically using algorithms based on differential inequalities [72, 58, 64, 66]. In this chapter, we aim to improve the accuracy of differential inequalities by using mean value enclosures rather than redundant constraints. However, we consider the constrained reachability problem in order to demonstrate how our mean value approach can be combined with the use of constraints whenever they are available.

We assume that the constraint set $G$ can be expressed as

$$G \equiv \left\{ (t, \mathbf{z}, \mathbf{p}) \in \mathbb{R}^{1+n_x+n_p} : \begin{array}{l} \mathbf{g}(t, \mathbf{z}, \mathbf{p}) \leq \mathbf{0} \\ \\ \mathbf{h}(t, \mathbf{z}, \mathbf{p}) = \mathbf{0} \end{array} \right\}, \tag{4.3}$$

where $(\mathbf{g}, \mathbf{h}) : D_G \subset \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_g} \times \mathbb{R}^{n_h}$ are locally Lipschitz continuous on $D_G$ and continuously differentiable with respect to $(\mathbf{z}, \mathbf{p})$ at each point in $D_G$. Since some technical details of our new method require redundant and non-redundant constraints to be treated differently, we further assume that the constraints can be partitioned into $\mathbf{g} = (\mathbf{g}^{\text{inv}}, \mathbf{g}^{\text{con}})$ and $\mathbf{h} = (\mathbf{h}^{\text{inv}}, \mathbf{h}^{\text{con}})$, where $\mathbf{g}^{\text{inv}}$ and $\mathbf{h}^{\text{inv}}$ denote *invariants* that are assumed to hold for all solutions; i.e.,

$$\left. \begin{array}{l} \mathbf{g}^{\text{inv}}(t, \mathbf{x}(t, \mathbf{p}), \mathbf{p}) \leq \mathbf{0} \\ \mathbf{h}^{\text{inv}}(t, \mathbf{x}(t, \mathbf{p}), \mathbf{p}) = \mathbf{0} \end{array} \right\}, \quad \forall (t, \mathbf{p}) \in I \times P, \tag{4.4}$$

and $\mathbf{g}^{\text{con}}$ and $\mathbf{h}^{\text{con}}$ denote conventional constraints that require no further assumptions.

## 4.3 Mean Value Differential Inequalities for Systems with Invariants Only

This section presents our first main result, Theorem 11, which provides a method for computing time-varying interval and mean-value enclosures of the solutions of (4.1). For technical reasons, we only make use of the invariants $\mathbf{g}^{\text{inv}}$ and $\mathbf{h}^{\text{inv}}$ in Theorem 11, and disregard the constraints $\mathbf{g}^{\text{con}}$ and $\mathbf{h}^{\text{con}}$. By (4.4), this provides valid time-varying bounds on $\mathbf{x}(t, \mathbf{p})$ for all $\mathbf{p} \in P$, rather than just for $\mathbf{p} \in P^*$. The extension to general constraints is taken up in Section 4.4.

**Definition 12** *Let* $\mathbf{s} : I \times P \to \mathbb{R}^{n_x \times n_p}$ *denote the first-order parametric sensitivity matrix for* (4.1a)*, defined by*

$$s_{ij}(t, \mathbf{p}) \equiv \frac{\partial x_i}{\partial p_j}(t, \mathbf{p}) \tag{4.5}$$

*for all* $i \in \{1, \ldots, n_x\}$ *and* $j \in \{1, \ldots, n_p\}$.

We use lower-case bold for the matrix $\mathbf{s}$ to avoid conflict with the use of capital letters for sets. When convenient, we denote $\left[\begin{smallmatrix}\mathbf{x}\\ \mathbf{s}\end{smallmatrix}\right]$ as the joint $(n_x + n_x n_p)$-dimensional vector of states and sensitivities by identifying $\mathbf{s}$ as a $n_x n_p$-dimensional vector formed by stacking its columns.

**Definition 13** *Define the functions* $\mathbf{s}_0 : P \to \mathbb{R}^{n_x \times n_p}$ *and* $\mathbf{f}_s : D_{f_s} \subset \mathbb{R} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_x \times n_p} \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_x \times n_p}$ *by*

$$\mathbf{s}_0(\mathbf{p}) \equiv \frac{\partial \mathbf{x}_0}{\partial \mathbf{p}}(\mathbf{p}), \tag{4.6}$$

$$\mathbf{f}_s(t, \mathbf{x}, \mathbf{s}, \mathbf{p}) \equiv \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(t, \mathbf{x}, \mathbf{p})\mathbf{s} + \frac{\partial \mathbf{f}}{\partial \mathbf{p}}(t, \mathbf{x}, \mathbf{p}). \tag{4.7}$$

With these definitions, the joint state and sensitivity vector $\left[\begin{smallmatrix}\mathbf{x}\\ \mathbf{s}\end{smallmatrix}\right]$ satisfy the following $n_x + n_x n_p$ ODEs:

$$\begin{aligned}\frac{d}{dt}\begin{bmatrix}\mathbf{x}(t,\mathbf{p})\\ \mathbf{s}(t,\mathbf{p})\end{bmatrix} &= \begin{bmatrix}\mathbf{f}(t,\mathbf{x}(t,\mathbf{p}),\mathbf{p})\\ \mathbf{f}_s(t,\mathbf{x}(t,\mathbf{p}),\mathbf{s}(t,\mathbf{p}),\mathbf{p})\end{bmatrix},\\ \begin{bmatrix}\mathbf{x}(t_0,\mathbf{p})\\ \mathbf{s}(t_0,\mathbf{p})\end{bmatrix} &= \begin{bmatrix}\mathbf{x}_0(\mathbf{p})\\ \mathbf{s}_0(\mathbf{p})\end{bmatrix}.\end{aligned} \tag{4.8}$$

We assume throughout that (4.8) has a unique solution on all of $I$ for every $\mathbf{p} \in P$.

Our new method relies on the relationship between $\mathbf{x}$ and $\mathbf{s}$ formalized by the following statements of the Mean Value Theorem (see Theorem 7.3 in [45] and Theorem 5.1.5 in [48], respectively).

**Theorem 10** *For any $i \in \{1,\ldots,n_x\}$, $t \in I$, and $\hat{\mathbf{p}}, \mathbf{p} \in P$, $\exists \boldsymbol{\xi} \in P$ such that*

$$x_i(t,\mathbf{p}) = x_i(t,\hat{\mathbf{p}}) + \mathbf{s}_i(t,\boldsymbol{\xi})(\mathbf{p}-\hat{\mathbf{p}}), \tag{4.9}$$

*where $\mathbf{s}_i$ denotes the $i^{th}$ row of $\mathbf{s}$.*

**Corollary 2** *Choose any $t \in I$ and let $S(t) \in \mathbb{IR}^{n_x \times n_p}$ satisfy $\mathbf{s}(t,\mathbf{p}) \in S(t)$, $\forall \mathbf{p} \in P$. For any $\hat{\mathbf{p}}, \mathbf{p} \in P$, $\exists \tilde{\mathbf{s}} \in S(t)$ such that*

$$\mathbf{x}(t,\mathbf{p}) = \mathbf{x}(t,\hat{\mathbf{p}}) + \tilde{\mathbf{s}}(\mathbf{p}-\hat{\mathbf{p}}). \tag{4.10}$$

*Furthermore,*

$$\mathbf{x}(t,\mathbf{p}) \in \mathbf{x}(t,\hat{\mathbf{p}}) + S(t)(\mathbf{p}-\hat{\mathbf{p}}), \quad \forall \mathbf{p} \in P. \tag{4.11}$$

Note that Theorem 10 and Corollary 2 depend on convexity of $P$, and Corollary 2 further relies on convexity of $S(t)$ (i.e., $\tilde{\mathbf{s}}$ need not lie in the true image set $\mathbf{s}(t, P)$).

Eq. (4.11) is a mean value enclosure of $\mathbf{x}$. By Corollary 2, this enclosure can be obtained by computing a valid pointwise-in-time interval enclosure of $\mathbf{s}(t, \mathbf{p})$. This can be done by simply applying standard differential inequalities to (4.8), which would also furnish an interval enclosure $X(t)$ of $\mathbf{x}(t, \mathbf{p})$. However, applying standard differential inequalities to (4.8) requires a method for computing bounds on the ranges of $\mathbf{f}$ and $\mathbf{f}_s$ over the current bounds $X(t) \times S(t) \times P$ at each $t$ (more precisely, over individual faces of this interval [64]). In the proposed approach, this basic scheme is improved by additionally using the relationship (4.10), along with any available invariants, to restrict the domains over which $\mathbf{f}$ and $\mathbf{f}_s$ must be bounded, ultimately leading to tighter bounds $X(t)$ and $S(t)$ and a tighter mean value enclosure (4.11).

Before developing this approach in detail, we first introduce one further constraint that can be used to restrict the domains over which $\mathbf{f}$ and $\mathbf{f}_s$ must be bounded. This constraint results from differentiating the second equation in (4.4) with respect to $\mathbf{p}$, which implies that

$$\frac{\partial \mathbf{h}^{\text{inv}}}{\partial \mathbf{x}}(t, \mathbf{x}(t, \mathbf{p}), \mathbf{p})\mathbf{s}(t, \mathbf{p}) + \frac{\partial \mathbf{h}^{\text{inv}}}{\partial \mathbf{p}}(t, \mathbf{x}(t, \mathbf{p}), \mathbf{p}) = \mathbf{0} \tag{4.12}$$

for all $(t, \mathbf{p}) \in I \times P$. Strictly, (4.4) only implies that (4.12) holds on the interior $\text{int}(P)$. However, provided that $\text{int}(P) \neq \emptyset$, (4.12) can be extended to the closure of $\text{int}(P)$ because the left-hand side is continuous w.r.t. $\mathbf{p}$, and since $P$ is an interval, the closure of $\text{int}(P)$ is exactly $P$. We assume henceforth that $\text{int}(P) \neq \emptyset$ without loss of generality since, if $P_j = [p_j^L, p_j^U]$ with $p_j^L = p_j^U$ for some $j$, then (4.1) can simply be restated with $p_j$ as a constant rather than an uncertain parameter.

To state our new bounding result generally, we next define a generic interval

operator $\mathcal{R}$. Conceptually, $\mathcal{R}$ can be thought of as any algorithm that takes intervals $P$, $X$, and $S$ as input (interpreted as bounds on $\mathbf{p}$, $\mathbf{x}(t,\mathbf{p})$, and $\mathbf{s}(t,\mathbf{p})$ at some fixed $t \in I$, respectively) and returns bounds on the possible values of the functions $\mathbf{f}$ and $\mathbf{f}_s$ that are achievable with arguments that are (i) contained in $P \times X \times S$, and (ii) consistent with the relations (4.10), (4.4), and (4.12). However, this simple conceptual description omits an important detail. In fact, $\mathcal{R}$ must take two sensitivity intervals as input, denoted by $S$ and $\tilde{S}$. The first represents a bound on $\mathbf{s}(t,\mathbf{p})$, while the second represents a bound on the variable $\tilde{\mathbf{s}}$ appearing in (4.10). This distinction is necessary because in general $\tilde{\mathbf{s}} \neq \mathbf{s}(t,\mathbf{p})$ in Corollary 2, and some details of our main bounding result rely on arguments about the possible values of $\mathbf{s}(t,\mathbf{p})$ in certain situations that do not apply to $\tilde{\mathbf{s}}$ (see further explanation after Theorem 11). In light of this issue, the properties required of $\mathcal{R}$ are stated precisely in the following definition.

**Definition 14** *Let* $\mathcal{R} : D_{\mathcal{R}} \subset \mathbb{R} \times \mathbb{IR}^{n_p} \times \mathbb{IR}^{n_x} \times \mathbb{IR}^{n_x \times n_p} \times \mathbb{IR}^{n_x \times n_p} \to \mathbb{IR}^{n_x + n_x n_p}$ *denote an interval refinement operator satisfying:*

1. *For any* $(t, P, X, S, \tilde{S}) \in D_{\mathcal{R}}$, $\mathcal{R}(t, P, X, S, \tilde{S})$ *is an interval containing the set*

$$\left\{ \begin{bmatrix} \boldsymbol{\sigma}_x \\ \boldsymbol{\sigma}_s \end{bmatrix} : \begin{array}{c} \boldsymbol{\sigma}_x = \mathbf{f}(t,\mathbf{x},\mathbf{p}) \\ \boldsymbol{\sigma}_s = \mathbf{f}_s(t,\mathbf{x},\mathbf{s},\mathbf{p}) \\ \mathbf{p} \in P,\ \mathbf{x} \in X,\ \mathbf{s} \in S,\ \tilde{\mathbf{s}} \in \tilde{S} \\ \mathbf{x} = \mathbf{x}(t,\hat{\mathbf{p}}) + \tilde{\mathbf{s}}(\mathbf{p} - \hat{\mathbf{p}}) \\ \mathbf{g}^{\mathrm{inv}}(t,\mathbf{x},\mathbf{p}) \leq \mathbf{0} \\ \mathbf{h}^{\mathrm{inv}}(t,\mathbf{x},\mathbf{p}) = \mathbf{0} \\ \frac{\partial \mathbf{h}^{\mathrm{inv}}}{\partial \mathbf{x}}(t,\mathbf{x},\mathbf{p})\mathbf{s} + \frac{\partial \mathbf{h}^{\mathrm{inv}}}{\partial \mathbf{p}}(t,\mathbf{x},\mathbf{p}) = \mathbf{0} \end{array} \right\},$$

*where* $\hat{\mathbf{p}} \in P$ *is a fixed reference point and is omitted from the argument list of* $\mathcal{R}$ *for brevity.*

*2. $\mathcal{R}$ is locally Lipschitz continuous.*

*3. $D_{\mathcal{R}}$ is open.*

A specific definition of $\mathcal{R}$ satisfying these properties is presented in Section 4.5. The statement of Theorem 11 also requires the following definitions.

**Definition 15** *For any interval $Z = [\mathbf{z}^L, \mathbf{z}^U] \in \mathbb{IR}^n$ and any $i \in \{1, \ldots, n\}$, define the interval* face selection *operators $\mathcal{B}_i^L, \mathcal{B}_i^U : \mathbb{IR}^n \to \mathbb{IR}^n$ and the interval bound selection operators $\pi_i^L, \pi_i^U : \mathbb{IR}^n \to \mathbb{R}$ by*

$$\mathcal{B}_i^L(Z) \equiv \{\mathbf{z} \in Z : z_i = z_i^L\}, \tag{4.13}$$

$$\mathcal{B}_i^U(Z) \equiv \{\mathbf{z} \in Z : z_i = z_i^U\}, \tag{4.14}$$

$$\pi_i^L(Z) \equiv z_i^L, \quad \pi_i^U(Z) \equiv z_i^U. \tag{4.15}$$

Definition 15 applies to interval matrices $S \in \mathbb{IR}^{n_x \times n_p}$ by identifying them with $n_x n_p$-dimensional interval vectors. Thus, $\mathcal{B}^{L/U}_{n_x(j-1)+i}(S)$ denotes the lower/upper face of the interval vector $S$ corresponding to the $(i,j)^{\text{th}}$ element of $S$. Below, we denote this set more simply by $\mathcal{B}_{ij}^{L/U}(S)$ and interpret it as an interval matrix.

**Theorem 11** *Let $\mathcal{R}$ satisfy Definition 14 with some reference point $\hat{\mathbf{p}} \in P$ and let $X_0(P) \in \mathbb{IR}^{n_x}$ and $S_0(P) \in \mathbb{IR}^{n_x \times n_p}$ satisfy $\mathbf{x}_0(\mathbf{p}) \in X_0(P)$ and $\mathbf{s}_0(\mathbf{p}) \in S_0(P)$, $\forall \mathbf{p} \in P$. Let $X(t) \equiv [\mathbf{x}^L(t), \mathbf{x}^U(t)] \in \mathbb{IR}^{n_x}$ and $S(t) \equiv [\mathbf{s}^L(t), \mathbf{s}^U(t)] \in \mathbb{IR}^{n_x \times n_p}$ be*

*solutions of the ODEs:*

$$
\begin{aligned}
\dot{x}_i^L(t) &= \pi_i^L \circ \mathcal{R}(t, P, \mathcal{B}_i^L(X(t)), S(t), S(t)), \\
\dot{x}_i^U(t) &= \pi_i^U \circ \mathcal{R}(t, P, \mathcal{B}_i^U(X(t)), S(t), S(t)), \\
\dot{s}_{ij}^L(t) &= \pi_{n_x+n_x(j-1)+i}^L \circ \mathcal{R}(t, P, X(t), \mathcal{B}_{ij}^L(S(t)), S(t)), \\
\dot{s}_{ij}^U(t) &= \pi_{n_x+n_x(j-1)+i}^U \circ \mathcal{R}(t, P, X(t), \mathcal{B}_{ij}^U(S(t)), S(t)), \\
X(t_0) &= X_0(P), \quad S(t_0) = S_0(P).
\end{aligned} \tag{4.16}
$$

*Then* $\mathbf{x}(t,\mathbf{p}) \in X(t)$ *and* $\mathbf{s}(t,\mathbf{p}) \in S(t)$, $\forall (t,\mathbf{p}) \in I \times P$. *Moreover,* $\mathbf{x}(t,\mathbf{p}) \in \mathbf{x}(t,\hat{\mathbf{p}}) + S(t)(\mathbf{p}-\hat{\mathbf{p}})$, $\forall (t,\mathbf{p}) \in I \times P$.

The proof of Theorem 11 involves several intermediate steps and is taken up in the Appendix. Regarding (4.16), note that the subscript $n_x + n_x(j-1) + i$ indexes the location of $s_{ij}$ in the vector $\left[\begin{smallmatrix}\mathbf{x}\\ \mathbf{s}\end{smallmatrix}\right]$ when $\mathbf{s}$ is vectorized by stacking its columns. Also note that $\mathcal{R}$ is always evaluated on individual faces of $X(t) \times S(t)$ in (4.16) due to the use of the face selection operators. The use of face selection operators is central to all bounding methods based on differential inequalities and arises from the observation that any trajectory $(\mathbf{x}(t,\mathbf{p}), \mathbf{s}(t,\mathbf{p}))$ that leaves the interval $X(t) \times S(t)$ must lie on its boundary at some point in time. Thus, to describe valid bounds, it is only necessary to consider the possible values that $\mathbf{f}$ and $\mathbf{f}_s$ can take on the faces of $X(t) \times S(t)$. However, the same argument can not be made for the value $\tilde{\mathbf{s}}$ in (4.10) since $\tilde{\mathbf{s}}$ need not equal $\mathbf{s}(t,\mathbf{p})$. This is the reason for defining $\mathcal{R}$ with an independent argument for a bound on $\tilde{\mathbf{s}}$, which is always the full interval $S(t)$ in (4.16). Once an algorithm for $\mathcal{R}$ is defined, the ODEs (4.16) can be solved numerically to yield valid bounds $X(t)$ and $S(t)$, along with a valid mean value enclosure of the form (4.11).

## 4.4 Mean Value Differential Inequalities for System with Invariants and Constraints

This section presents our second main result, Theorem 12, which extends Theorem 11 to make use of the constraints $\mathbf{g}^{\text{con}}$ and $\mathbf{h}^{\text{con}}$ in addition to the invariants $\mathbf{g}^{\text{inv}}$ and $\mathbf{h}^{\text{inv}}$. As a result, Theorem 12 provides time-varying bounds on $\mathbf{x}(t, \mathbf{p})$ for all $\mathbf{p} \in P^*$, but not necessarily for all $\mathbf{p} \in P$. At first glance, it may seem that this can be achieved by simply modifying Definition 14 so that the set that must be enclosed by $\mathcal{R}(t, P, X, S, \tilde{S})$ in Condition 1 includes the additional constraints $\mathbf{g}^{\text{con}}$ and $\mathbf{h}^{\text{con}}$. However, this proves to be invalid on account of a technical conflict between the constraints $\mathbf{g}^{\text{con}}$ and $\mathbf{h}^{\text{con}}$ and the mean value relation (4.10), which is also used in Condition 1. Specifically, since the bounding ODEs for $X(t)$ and $S(t)$ are coupled, any method that produces bounds $X(t)$ that only enclose $\mathbf{x}(t, \mathbf{p})$ for $\mathbf{p} \in P^*$ will also produce bounds $S(t)$ that only enclose $\mathbf{s}(t, \mathbf{p})$ for $\mathbf{p} \in P^*$. However, by Corollary 2, the interval $S(t)$ must enclose $\mathbf{s}(t, \mathbf{p})$ for all $\mathbf{p} \in P$, not just $\mathbf{p} \in P^*$, in order to be used to bound the variable $\tilde{\mathbf{s}}$ in the constraint $\mathbf{x} = \mathbf{x}(t, \hat{\mathbf{p}}) + \tilde{\mathbf{s}}(\mathbf{p} - \hat{\mathbf{p}})$ in Condition 1 of Definition 14, as well as to be used in the final mean value inclusion (4.11). Moreover, Corollary 2 cannot be restated with $P^*$ in place of $P$ because $P^*$ may not be convex.

To avoid this conflict, we propose a two step procedure wherein Theorem 11 is first applied to obtain bounds $X(t)$ and $S(t)$ that are valid for all $\mathbf{p} \in P$. Next, a second bounding computation is done to compute refined bounds $X^*(t)$ that are valid only for $\mathbf{p} \in P^*$. This second step makes use of $S(t)$ to avoid the issue outlined above, while $X(t)$ is a byproduct that is simply discarded. Although this two step procedure is clearly less efficient than a single bounding computation, the second step does not require new bounds to be computed for $\mathbf{s}(t, \mathbf{p})$, so the system of bounding ODEs is significantly smaller. To state the second step in detail, the

following modified refinement operator is required. Recall that $\mathbf{h}$ and $\mathbf{g}$ denote the full constraint vectors $\mathbf{g} = (\mathbf{g}^{\text{inv}}, \mathbf{g}^{\text{con}})$ and $\mathbf{h} = (\mathbf{h}^{\text{inv}}, \mathbf{h}^{\text{con}})$.

**Definition 16** *Let $\mathcal{R}^* : D_{\mathcal{R}^*} \subset \mathbb{R} \times \mathbb{IR}^{n_p} \times \mathbb{IR}^{n_x} \times \mathbb{IR}^{n_x \times n_p} \to \mathbb{IR}^{n_x}$ denote an interval refinement operator satisfying:*

1. *For any $(t, P, X, S) \in D_{\mathcal{R}^*}$, $\mathcal{R}^*(t, P, X, S)$ contains the set*

$$\left\{ \boldsymbol{\sigma}_x : \begin{array}{c} \boldsymbol{\sigma}_x = \mathbf{f}(t, \mathbf{x}, \mathbf{p}) \\ \mathbf{p} \in P,\ \mathbf{x} \in X,\ \tilde{\mathbf{s}} \in S \\ \mathbf{x} = \mathbf{x}(t, \hat{\mathbf{p}}) + \tilde{\mathbf{s}}(\mathbf{p} - \hat{\mathbf{p}}) \\ \mathbf{g}(t, \mathbf{x}, \mathbf{p}) \leq \mathbf{0},\ \mathbf{h}(t, \mathbf{x}, \mathbf{p}) = \mathbf{0} \end{array} \right\},$$

   *where $\hat{\mathbf{p}} \in P$ is a fixed reference point and is omitted from the argument list of $\mathcal{R}^*$ for brevity.*

2. *$\mathcal{R}^*$ is locally Lipschitz continuous.*

3. *$D_{\mathcal{R}^*}$ is open.*

**Theorem 12** *Let $S : I \to \mathbb{IR}^{n_x \times n_p}$ be a locally Lipschitz continuous function satisfying $\mathbf{s}(t, \mathbf{p}) \in S(t)$, $\forall (t, \mathbf{p}) \in I \times P$. Let $\mathcal{R}^*$ satisfy Definition 16 with some reference point $\hat{\mathbf{p}} \in P$ and let $X_0(P) \in \mathbb{IR}^{n_x}$ satisfy $\mathbf{x}_0(\mathbf{p}) \in X_0(P)$, $\forall \mathbf{p} \in P$. Let $X^*(t) \equiv [\mathbf{x}^{*,L}(t), \mathbf{x}^{*,U}(t)] \in \mathbb{IR}^{n_x}$ be a solution of the ODEs:*

$$\begin{aligned} \dot{x}_i^{*,L}(t) &= \pi_i^L \circ \mathcal{R}^*(t, P, \mathcal{B}_i^L(X^*(t)), S(t)), \\ \dot{x}_i^{*,U}(t) &= \pi_i^U \circ \mathcal{R}^*(t, P, \mathcal{B}_i^U(X^*(t)), S(t)), \\ X^*(t_0) &= X_0(P). \end{aligned} \tag{4.17}$$

*Then $\mathbf{x}(t, \mathbf{p}) \in X^*(t)$ for all $(t, \mathbf{p}) \in I \times P^*$.*

The proof of Theorem 12 is taken up after the proof of Theorem 11 in the Appendix. Given appropriate algorithms for $\mathcal{R}$ and $\mathcal{R}^*$, Theorem 12 is implemented by numerically solving the ODEs (4.16) to obtain $S(t)$, and then solving (4.17) for $X^*(t)$. In practice, (4.16) and (4.17) can be solved simultaneously to avoid storing $S(t)$. Note that continuity of the interval function $\mathcal{R}$ implies that the right-hand sides of the ODEs in (4.16) are continuous, and it follows that the solutions $s_{ij}^{L/U}(t)$ are locally Lipschitz continuous functions of $t$. Thus, the solution of (4.16) satisfies the hypotheses on $S(t)$ required by Theorem 12.

## 4.5 An Algorithm for the Refinement Operator $\mathcal{R}$

This section provides an algorithm for the refinement operator $\mathcal{R}$ satisfying Definition 14. An algorithm for $\mathcal{R}^*$ can be developed analogously and is omitted for brevity. The following assumption can be satisfied using natural interval extensions as discussed in detail in §5 in [66].

**Assumption 7** *Locally Lipschitz continuous inclusion functions are available for* $\mathbf{f}$, $\frac{\partial \mathbf{f}}{\partial \mathbf{x}}$, $\frac{\partial \mathbf{f}}{\partial \mathbf{p}}$, $\mathbf{g}$, $\frac{\partial \mathbf{g}}{\partial \mathbf{x}}$, $\frac{\partial \mathbf{g}}{\partial \mathbf{p}}$, $\mathbf{h}$, $\frac{\partial \mathbf{h}}{\partial \mathbf{x}}$, *and* $\frac{\partial \mathbf{h}}{\partial \mathbf{p}}$ *on an open set* $D_\square \subset \mathbb{I}D_f \cap \mathbb{I}D_G$. *These are denoted by square brackets; e.g.,* $[\mathbf{f}] : D_\square \to \mathbb{IR}^{n_x}$, $\left[\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right] : D_\square \to \mathbb{IR}^{n_x \times n_x}$, *etc. Furthermore,* $D_\square$ *has the property that if* $(T, X, P) \in D_\square$ *then* $\mathbb{I}T \times \mathbb{I}X \times \mathbb{I}P \subset D_\square$.

As per Definition 14, $\mathcal{R}$ has two key functions. First, given $(t, P, X, S, \tilde{S}) \in D_{\mathcal{R}}$, $\mathcal{R}$ refines the intervals $(P, X, S, \tilde{S})$ by eliminating regions where the mean value relation (4.10) or the invariants (4.4) and (4.12) are violated. Second, $\mathcal{R}$ bounds the ranges of $\mathbf{f}$ and $\mathbf{f}_s$ over the refined domain. To describe the refinement step, choose any $(t, P, X, S, \tilde{S}) \in D_{\mathcal{R}}$, let $\hat{\mathbf{p}} \in P$ be the reference point used in Definition 14, and define the shorthand $\hat{\mathbf{x}} = \mathbf{x}(t, \hat{\mathbf{p}})$. We first consider the mean value relation, which is

given componentwise by

$$x_i = \hat{x}_i + \tilde{\mathbf{s}}_i(\mathbf{p} - \hat{\mathbf{p}}), \tag{4.18}$$

where $\tilde{\mathbf{s}}_i$ denotes the $i^{\text{th}}$ row of $\tilde{\mathbf{s}}$. An updated interval bound $X_i$ can be obtained by evaluating the right-hand side (r.h.s.) of (4.18) in interval arithmetic over $P \times \tilde{S}$. A similar update for each $P_j$ and $\tilde{S}_{ij}$ can be achieved by considering rearrangements of (4.18) that isolate each of these variables. However, this may involve division by intervals containing zero, and common methods for dealing with this such as extended interval arithmetic would cause $\mathcal{R}$ to violate the Lipschitz property required by Definition 14. Instead, we multiply (4.18) by a constant $\mu$ and add $\tilde{s}_{ij}$ on both sides to obtain

$$\tilde{s}_{ij} = \mu(\hat{x}_i - x_i + \tilde{\mathbf{s}}_i(\mathbf{p} - \hat{\mathbf{p}})) + \tilde{s}_{ij}. \tag{4.19}$$

Collecting the $\tilde{s}_{ij}$ terms on the right gives,

$$\begin{aligned} \tilde{s}_{ij} =& \mu\big(\hat{x}_i - x_i + \sum_{k \neq j} \tilde{s}_{ik}(p_k - \hat{p}_k)\big) + \\ & (1 + \mu(p_j - \hat{p}_j))\tilde{s}_{ij}, \end{aligned} \tag{4.20}$$

which can be used to update $\tilde{S}_{ij}$ by evaluating the r.h.s. in interval arithmetic. We apply this refinement for every $\tilde{S}_{ij}$, and an analogous refinement for every $P_j$. The choice of $\mu$ should minimize the conservatism introduced by the term $(1+\mu(p_j - \hat{p}_j))\tilde{s}_{ij}$. We use two different values of $\mu$, $\mu^{\pm} = \pm \max(\epsilon, |P_j - \hat{p}_j|)$, where $\epsilon > 0$ is a user-defined tolerance.

To refine $(P, X, S)$ further based on (4.4) and (4.12), Algorithm 1 in [66]

can be used with $Z = (X, S)$ (more specifically, only lines 3-24 should be used). For linear constraints, Algorithm 1 in [67] can be used instead and may give better results. We omit the details here for brevity and denote this refinement simply by $(P, X, S) \leftarrow \mathcal{I}_G(t, P, X, S)$. Under Assumption 7, Theorem 2 in [66] ensures that $\mathcal{I}_G$ is defined and locally Lipschitz continuous on the set of inputs $(t, P, X, S)$ such that $([t, t], X, P) \in D_\square$.

Let $(X^\dagger, P^\dagger, S^\dagger, \tilde{S}^\dagger)$ denote intervals resulting from the refinements above. Next, we discuss bounding the ranges of $\mathbf{f}$ and $\mathbf{f}_s$ over this refined domain. The range of $\mathbf{f}_s(t, \cdot, \cdot, \cdot)$ is bounded over $X^\dagger \times S^\dagger \times P^\dagger$ using the inclusion function

$$[\mathbf{f}_s](t, X^\dagger, S^\dagger, P^\dagger) \equiv \left[\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right](t, X^\dagger, P^\dagger) S^\dagger + \left[\frac{\partial \mathbf{f}}{\partial \mathbf{p}}\right](t, X^\dagger, P^\dagger). \tag{4.21}$$

The range of $\mathbf{f}(t, \cdot, \cdot)$ can similarly be bounded over $X^\dagger \times P^\dagger$ by $[\mathbf{f}](t, X^\dagger, P^\dagger)$. However, Condition 1 of Definition 14 permits $\mathbf{f}(t, \cdot, \cdot)$ to be bounded over the subset

$$\{(\mathbf{x}, \mathbf{p}) \in X^\dagger \times P^\dagger : \mathbf{x} = \hat{\mathbf{x}} + \tilde{\mathbf{s}}(\mathbf{p} - \hat{\mathbf{p}}),\ \tilde{\mathbf{s}} \in \tilde{S}^\dagger\}, \tag{4.22}$$

which makes use of the mean value enclosure of $\mathbf{x}$. To do so, choose any $i \in \{1, \ldots, n_x\}$ and any $(\mathbf{x}, \mathbf{p})$ in (4.22). Applying the Mean Value Theorem to $f_i(t, \mathbf{x}, \mathbf{p})$ w.r.t. $\mathbf{p}$ ensures that $\exists \boldsymbol{\zeta}$ lying between $\mathbf{p}$ and $\hat{\mathbf{p}}$ such that

$$f_i(t, \mathbf{x}, \mathbf{p}) = f_i(t, \mathbf{x}, \hat{\mathbf{p}}) + \frac{\partial f_i}{\partial \mathbf{p}}(t, \mathbf{x}, \boldsymbol{\zeta})(\mathbf{p} - \hat{\mathbf{p}}). \tag{4.23}$$

Next, we apply the Mean Value Theorem to the term $f_i(t, \mathbf{x}, \hat{\mathbf{p}})$ w.r.t. $x_j$, $\forall j \neq i$. Although $f_i(t, \mathbf{x}, \hat{\mathbf{p}})$ could be expanded with respect to all $x_j$, the idea here is to

exploit the fact that, in the bounding ODEs (4.16), whenever the output of $\mathcal{R}$ is used to bound $f_i$ specifically, the $X$-argument to $\mathcal{R}$ has zero width in the $i^{\text{th}}$ dimension. It follows that $X^\dagger$ will also have $w(X_i^\dagger) = 0$. Thus, while it is necessary to develop an algorithm for $\mathcal{R}$ that is well-defined for all inputs $(t, P, X, S, \tilde{S}) \in D_{\mathcal{R}}$, we are at liberty to bound each $f_i$ in a way that will be particularly effective when $w(X_i^\dagger) = 0$.

For any $\boldsymbol{\gamma} \in \mathbb{R}^{n_x}$, let $\boldsymbol{\gamma}_{\not i} \equiv (\gamma_1, \cdots, \gamma_{i-1}, \gamma_{i+1}, \cdots, \gamma_{n_x})$. With a slight abuse of notation, denote $f_i(t, \boldsymbol{\gamma}_{\not i}, \gamma_i, \mathbf{p}) = f_i(t, \boldsymbol{\gamma}, \mathbf{p})$. With the reference state $\hat{\mathbf{x}}$ defined above, the Mean Value Theorem ensures that there exists $\boldsymbol{\gamma}_{\not i}$ lying between $\mathbf{x}_{\not i}$ and $\hat{\mathbf{x}}_{\not i}$ such that

$$\begin{aligned} f_i(t, \mathbf{x}, \mathbf{p}) = f_i(t, \hat{\mathbf{x}}_{\not i}, x_i, \hat{\mathbf{p}}) + \frac{\partial f_i}{\partial \mathbf{p}}(t, \mathbf{x}, \boldsymbol{\zeta})(\mathbf{p} - \hat{\mathbf{p}}) \\ + \sum_{k \neq i} \frac{\partial f_i}{\partial x_k}(t, \boldsymbol{\gamma}_{\not i}, x_i, \hat{\mathbf{p}})(x_k - \hat{x}_k). \end{aligned} \tag{4.24}$$

But since $(\mathbf{x}, \mathbf{p})$ is in (4.22), $\exists \tilde{\mathbf{s}} \in \tilde{S}^\dagger$ such that

$$\begin{aligned} &f_i(t, \mathbf{x}, \mathbf{p}) = f_i(t, \hat{\mathbf{x}}_{\not i}, x_i, \hat{\mathbf{p}}) + \\ &\sum_{j=1}^{n_p} \Big( \sum_{k \neq i}^{n_x} \frac{\partial f_i}{\partial x_k}(t, \boldsymbol{\gamma}_{\not i}, x_i, \hat{\mathbf{p}}) \tilde{s}_{kj} + \frac{\partial f_i}{\partial p_j}(t, \mathbf{x}, \boldsymbol{\zeta}) \Big)(p_j - \hat{p}_j). \end{aligned} \tag{4.25}$$

Since we are defining $\mathcal{R}$ for an arbitrary argument $(t, P, X, S, \tilde{S}) \in D_{\mathcal{R}}$, there is no guarantee that $\hat{\mathbf{x}} \in X$. It is therefore possible that, after refinement, $(\hat{\mathbf{x}}, \hat{\mathbf{p}}) \notin X^\dagger \times P^\dagger$. Thus, we cannot conclude that $\boldsymbol{\zeta} \in P^\dagger$ and $\boldsymbol{\gamma}_{\not i} \in X_{\not i}^\dagger$, despite convexity of $P^\dagger$ and $X_{\not i}^\dagger$. However, letting $\square(A)$ denote the interval hull of a set $A$ and defining $\bar{P}^\dagger \equiv \square(P^\dagger \cup \{\hat{\mathbf{p}}\})$ and $\bar{X}^\dagger \equiv \square(X^\dagger \cup \{\hat{\mathbf{x}}\})$, it follows that $\boldsymbol{\zeta} \in \bar{P}^\dagger$ and $\boldsymbol{\gamma}_{\not i} \in \bar{X}_{\not i}^\dagger$. Therefore,

by Assumption 7,

$$f_i(t,\mathbf{x},\mathbf{p}) \in [f_i](t,\hat{\mathbf{x}}_\ell, X_i^\dagger, \hat{\mathbf{p}}) + \sum_{j=1}^{n_p}\left(\sum_{k\neq i}^{n_x}\left[\frac{\partial f_i}{\partial x_k}\right](t,\bar{X}_\ell^\dagger, X_i^\dagger, \hat{\mathbf{p}})\tilde{S}_{kj}^\dagger + \left[\frac{\partial f_i}{\partial p_j}\right](t,X^\dagger,\bar{P}^\dagger)\right)(P_j^\dagger - \hat{p}_j). \tag{4.26}$$

We use the shorthand $F_{MV}(t,P^\dagger,X^\dagger,\tilde{S}^\dagger)$ to denote the interval vector whose $i^{\text{th}}$ element is the r.h.s. of (4.26). Since $(\mathbf{x},\mathbf{p})$ was chosen arbitrarily from the set (4.22), we have shown that $\mathbf{f}(t,\mathbf{x},\mathbf{p}) \in F_{MV}(t,P^\dagger,X^\dagger,\tilde{S}^\dagger)$ for all $(\mathbf{x},\mathbf{p})$ in (4.22), as desired. In Algorithm 4 below, we bound $\mathbf{f}$ on (4.22) using both $F_{MV}(t,P^\dagger,X^\dagger,\tilde{S}^\dagger)$ and the inclusion function $[\mathbf{f}](t,X^\dagger,P^\dagger)$.

The complete algorithm for $\mathcal{R}$ is given in Algorithm 4. Throughout the algorithm, primed variables are used to temporarily store updated bounds. The notation $\bar{\cap}$ denotes the extended intersection $Y\bar{\cap}Z \equiv [\text{mid}(z^L,y^L,y^U),\text{mid}(z^U,y^L,y^U)]$, where $\text{mid}(a,b,c)$ denotes the middle value of $a,b,c\in\mathbb{R}$. Note that $Y\bar{\cap}Z$ agrees with $Y\cap Z$ whenever $Y\cap Z$ is nonempty, and is a nonempty subinterval of $Y$ otherwise.

**Theorem 13** *Let $D_{\mathcal{R}}$ be the set of all $(t,P,X,S,\tilde{S})$ in $\mathbb{R}\times\mathbb{IR}^{n_p}\times\mathbb{IR}^{n_x}\times\mathbb{IR}^{n_x\times n_p}\times\mathbb{IR}^{n_x\times n_p}$ such that $([t,t],\square(X\cup\mathbf{x}(t,\hat{\mathbf{p}})),\square(P\cup\hat{\mathbf{p}}))\in D_\square$. The operator $\mathcal{R}$ defined by Algorithm 4 is well-defined for every $(t,P,X,S,\tilde{S})\in D_{\mathcal{R}}$ and satisfies Definition 14.*

**Proof 1** *Choose any $(t,P,X,S,\tilde{S})\in D_{\mathcal{R}}$ and define $\bar{P}\equiv\square(P\cup\hat{\mathbf{p}})$ and $\bar{X}=\square(X\cup\mathbf{x}(t,\hat{\mathbf{p}}))$. By the definition of $D_{\mathcal{R}}$, we have*

$$([t,t],\bar{X},\bar{P})\in D_\square \quad\text{and}\quad ([t,t],X,P)\in D_\square, \tag{4.27}$$

*where the second inclusion follows from Assumption 7 and the fact that $X\subset\bar{X}$ and*

**Algorithm 4** An implementation of $\mathcal{R}$

1: **function** $\mathcal{R}(t, P, X, S, \tilde{S})$
2: **for** $i \leftarrow 1$ to $n_x$ **do**
3: $X_i' \leftarrow \hat{x}_i + \sum_{k=1}^{n_p} \tilde{S}_{ik}(P_k - \hat{p}_k)$
4: $X_i \leftarrow X_i \bar{\cap} X_i'$
5: **for** $j \leftarrow 1$ to $n_p$ **do**
6: $\mu \leftarrow 1/\max(\epsilon, |\tilde{S}_{ij}|)$
7: $\alpha \leftarrow \hat{x}_i - X_i + \sum_{k \neq j} \tilde{S}_{ik}(P_k - \hat{p}_k)$
8: $P_j' \leftarrow \mu\alpha + (1 + \mu\tilde{S}_{ij})(P_j - \hat{p}_j) + \hat{p}_j$
9: $P_j \leftarrow P_j \bar{\cap} P_j'$
10: $\mu \leftarrow -\mu$ and repeat lines 8-9
11: $\mu \leftarrow 1/\max(\epsilon, |P_j - \hat{p}_j|)$
12: $\tilde{S}_{ij}' \leftarrow \mu\alpha + (1 + \mu(P_j - \hat{p}_j))\tilde{S}_{ij}$
13: $\tilde{S}_{ij} \leftarrow \tilde{S}_{ij} \bar{\cap} \tilde{S}_{ij}'$
14: $\mu \leftarrow -\mu$ and repeat lines 12-13
15: **end for** ▷ $j \leftarrow 1$ to $n_p$
16: **end for** ▷ $i \leftarrow 1$ to $n_x$
17: $(P, X, S) \leftarrow \mathcal{I}_G(t, P, X, S)$
18: $\Sigma_s \leftarrow [\mathbf{f}_s](t, X, S, P)$
19: $\Sigma_x \leftarrow [\mathbf{f}](t, X, P) \cap F_{MV}(t, P, X, \tilde{S})$
20: **return** $(\Sigma_x, \Sigma_s)$
21: **end function**

$P \subset \bar{P}$. *Assumption 7 further implies that any subintervals* $P' \subset P$ *and* $X' \subset X$ *also satisfy* $([t,t], X', P') \in D_{\square}$ *and* $([t,t], \bar{X}', \bar{P}') \in D_{\square}$, *where* $\bar{X}' = \square(X' \cup \mathbf{x}(t, \hat{\mathbf{p}}))$ *and* $\bar{P}' \equiv \square(P' \cup \hat{\mathbf{p}})$. *Since Algorithm 4 only ever overwrites* $X$ *and* $P$ *with smaller refined intervals, this implies that* (4.27) *holds with the values currently stored in* $X$ *and* $P$ *at any point in the algorithm.*

*All of the interval operations in Algorithm 4 prior to line 17 are well defined for any arguments. As discussed above,* (4.27) *implies that the refinement* $\mathcal{I}_G(t, P, X, S)$ *in line 17 is well-defined. Equation* (4.27) *further ensures that* $[\mathbf{f}_s](t, X, S, P)$ *and* $[\mathbf{f}](t, X, P)$ *on lines 18–19 are well-defined. The inclusion functions for* $\mathbf{f}$ *and its partial derivatives used to evaluate* $F_{MV}(t, P, X, \tilde{S})$ *in* (4.26) *are also well-defined, which follows from* (4.27) *and Assumption 7 because the* $X$ *and* $P$ *arguments to all of these functions are subsets of* $\bar{X}$ *and* $\bar{P}$. *Therefore,* $\mathcal{R}$ *is well defined on* $D_{\mathcal{R}}$.

*Condition* 1 *of Definition* 14 *follows directly from* (4.20), (4.26), *and the inclusion properties of* $[\mathbf{f}]$, $[\mathbf{f}_s]$, *and* $\mathcal{I}_G$.

*To verify Condition* 2 *of Definition* 14, *we argue that every line of Algorithm 4 defines its output as a locally Lipschitz continuous function of* $t$ *and the current value of* $(P, X, S, \tilde{S})$. *Thus, Algorithm 4 defines* $\mathcal{R}$ *as a finite composition of locally Lipschitz functions, and it follows that* $\mathcal{R}$ *is locally Lipschitz continuous with respect to the input* $(t, P, X, S, \tilde{S}) \in D_{\mathcal{R}}$. *Interval addition and multiplication are locally Lipschitz continuous functions [48], as is the extended intersection* $\bar{\cap}$ *[61]. Moreover, it is straightforward to show that the mapping* $\mathbb{IR} \ni Q \mapsto 1/\max(\epsilon, |Q|) \in \mathbb{R}$ *is Lipschitz continuous with constant* $\epsilon^2$. *Thus, all of the operations in lines 2–16 are locally Lipschitz continuous. By Theorem 2 in [66], the same is true of the function* $\mathcal{I}_G$ *in line 17. Assumption 7 and the local Lipschitz continuity of interval multiplication imply that* $[\mathbf{f}_s]$ *in line 18 is locally Lipschitz. The same is true of* $[\mathbf{f}]$ *in line 19. Thus, it only remains to show that* $F_{MV}$ *is locally Lipschitz w.r.t.* $(t, P, X, \tilde{S})$, *which follows*

*from* (4.26) *given that the mapping* $P \mapsto \square(P, \hat{\mathbf{p}})$ *is Lipschitz w.r.t.* $P$, $(t, X) \mapsto \square(X, \mathbf{x}(t, \hat{\mathbf{p}}))$ *is locally Lipschitz w.r.t.* $(t, X)$, *the functions* $[f_i]$, $\left[\frac{\partial f_i}{\partial x_k}\right]$ *and* $\left[\frac{\partial f_i}{\partial p_j}\right]$ *are all locally Lipschitz by Assumption 7, and the interval additions and multiplications in* (4.26) *are locally Lipschitz as per [48].*

*Finally, Condition* 3 *of Definition 14 holds because* $D_\square$ *is open by Assumption 7 and* $D_\mathcal{R}$ *is defined as the inverse image of* $D_\square$ *under the continuous mapping* $(t, P, X, S, \tilde{S}) \mapsto ([t, t], \square(X \cup \mathbf{x}(t, \hat{\mathbf{p}})), \square(P \cup \hat{\mathbf{p}}))$. □

**Remark 3** *Algorithm 4 is the most straightforward implementation of the methods developed in this section. Several important modifications that significantly reduce the computational complexity and may result in tighter bounds are discussed in Appendix 4.8.1. These modifications are used in all numerical experiments in §4.7.*

## 4.6 Second Order Convergence Rate of MVDI

In this section, we prove that the enclosures provided by MVDI converge to the true reachable set at a quadratic rate as the width of $P$ tends towards zero. In contrast, standard differential inequalities (DI) has only first-order convergence [56]. Second-order convergence is highly desirable in algorithms that rely on partitioning $P$ to obtain tighter enclosures, as is common in bounded error estimation, global optimization, and constraint satisfaction problems [54, 22]. To begin, the following lemma extends the first-order convergence property of standard DI to the interval bounds obtained via Theorem 11.

**Lemma 3** *Let* $\bar{P} \in \mathbb{IR}^{n_p}$ *and let* $X_0 : \mathbb{I}\bar{P} \to \mathbb{IR}^{n_x}$ *and* $S_0 : \mathbb{I}\bar{P} \to \mathbb{IR}^{n_x \times n_p}$ *be inclusion*

*functions for* $\mathbf{x}_0$ *and* $\mathbf{s}_0$*,respectively. Assume* $\exists \lambda_0 \geq 0$ *such that*

$$w\left(\left[\begin{smallmatrix} X_0(P) \\ S_0(P) \end{smallmatrix}\right]\right) \leq \lambda_0 w(P), \quad \forall P \in \mathbb{I}\bar{P}. \tag{4.28}$$

*Let* $\mathcal{R}$ *satisfy Definition 14 and assume that* (4.16) *has solutions* $X_P(t)$ *and* $S_P(t)$ *for every* $P \in \mathbb{I}\bar{P}$*. Assume that, given any compact* $K \subset D_{\mathcal{R}}$*,* $\exists \lambda_{\mathcal{R}} \geq 0$ *such that*

$$w(\mathcal{R}(t, P, X, S, \tilde{S})) \leq \lambda_{\mathcal{R}} w\left(\left[\begin{smallmatrix} P \\ X \\ S \\ \tilde{S} \end{smallmatrix}\right]\right) \tag{4.29}$$

*for all* $(t, P, X, S, \tilde{S}) \in K$*. Then, for every* $t \in I$*,* $\exists \lambda \geq 0$ *such that*

$$w\left(\left[\begin{smallmatrix} X_P(t) \\ S_P(t) \end{smallmatrix}\right]\right) \leq \lambda w(P), \quad \forall P \in \mathbb{I}\bar{P}. \tag{4.30}$$

**Proof 2** *For all* $(i, j) \in \{1, \ldots, n_x\} \times \{1, \ldots, n_p\}$*, let*

$$K_i^L \equiv \{(t, P, \mathcal{B}_i^L(X_P(t)), S_P(t), S_P(t)) : t \in I,\ P \in \mathbb{I}\bar{P}\}$$
$$K_{ij}^L \equiv \{(t, P, X_P(t), \mathcal{B}_{ij}^L(S_P(t)), S_P(t)) : t \in I,\ P \in \mathbb{I}\bar{P}\}$$

*and define* $K_i^U$ *and* $K_{ij}^U$ *analogously. Since* $X_P(t)$ *and* $S_P(t)$ *are solutions of* (4.16)*, these sets must lie in* $D_{\mathcal{R}}$ *for all* $(i, j)$*. Moreover, these sets are compact in the metric space* $I \times \mathbb{IR}^{n_p} \times \mathbb{IR}^{n_x} \times \mathbb{IR}^{n_x \times n_p} \times \mathbb{IR}^{n_x \times n_p}$*. Let* $K \equiv \cup_{i,j}(K_i^L \cup K_i^U \cup K_{ij}^L \cup K_{ij}^U)$*, and let* $\lambda_{\mathcal{R}}$ *satisfy* (4.29)*. Moreover, let* $L \geq 0$ *be a Lipschitz constant for* $\mathcal{R}$ *on* $K$*, which exists by local Lipschitz continuity of* $\mathcal{R}$*.*

*For every $\tau \in I$, introduce the shorthand*

$$\begin{aligned}
\mathcal{R}_i^L(\tau) &= \pi_i^L \circ \mathcal{R}(\tau, P, \mathcal{B}_i^L(X_P(\tau)), S_P(\tau), S_P(\tau)), \\
\mathcal{R}_i^U(\tau) &= \pi_i^U \circ \mathcal{R}(\tau, P, \mathcal{B}_i^U(X_P(\tau)), S_P(\tau), S_P(\tau)), \\
\mathcal{R}_i^{UL}(\tau) &= \pi_i^U \circ \mathcal{R}(\tau, P, \mathcal{B}_i^L(X_P(\tau)), S_P(\tau), S_P(\tau)).
\end{aligned}$$

*For any $i$, the integral form of the ODEs* (4.16) *implies*

$$w(X_{P,i}(t)) = w(X_{P,i}(t_0)) + \int_{t_0}^{t} \mathcal{R}_i^U(\tau) - \mathcal{R}_i^L(\tau) d\tau. \tag{4.31}$$

*Using the* (4.29) *and the Lipschitz condition on $\mathcal{R}$, the integrand can be bounded pointwise:*

$$\begin{aligned}
&\mathcal{R}_i^U(\tau) - \mathcal{R}_i^L(\tau) \\
&= \left(\mathcal{R}_i^U(\tau) - \mathcal{R}_i^{UL}(\tau)\right) + \left(\mathcal{R}_i^{UL}(\tau) - \mathcal{R}_i^L(\tau)\right), \\
&\leq L d_H(\mathcal{B}_i^U(X_P(\tau)), \mathcal{B}_i^L(X_P(\tau))) + \\
&\qquad w(\mathcal{R}(\tau, P, \mathcal{B}_i^L(X_P(\tau)), S_P(\tau), S_P(\tau))), \\
&\leq L w(X_{P,i}(\tau)) + \lambda_{\mathcal{R}} w\left(\begin{bmatrix} P \\ \mathcal{B}_i^L(X_P(\tau)) \\ S_P(\tau) \end{bmatrix}\right), \\
&\leq \lambda_{\mathcal{R}} w(P) + (L + \lambda_{\mathcal{R}}) w\left(\begin{bmatrix} X_P(\tau) \\ S_P(\tau) \end{bmatrix}\right).
\end{aligned} \tag{4.32}$$

*Combining* (4.31), (4.32), *and* (4.28),

$$\begin{aligned}
w(X_{P,i}(t)) &\leq \lambda_0 w(P) + \lambda_{\mathcal{R}}(t_f - t_0) w(P) \\
&+ \int_{t_0}^{t} (L + \lambda_{\mathcal{R}}) w\left(\begin{bmatrix} X_P(\tau) \\ S_P(\tau) \end{bmatrix}\right) d\tau.
\end{aligned} \tag{4.33}$$

*An analogous argument shows that the right-hand side of* (4.33) *is also an upper*

*bound on* $w(S_{P,ij}(t))$ *for all* $(i,j)$*, and hence on* $w\left(\left[\begin{smallmatrix} X_P(t) \\ S_P(t) \end{smallmatrix}\right]\right)$*. Then, by Gronwall's inequality (Lemma 2.1 in [56]),*

$$w\left(\left[\begin{smallmatrix} X_P(t) \\ S_P(t) \end{smallmatrix}\right]\right) \leq (\lambda_0 + \lambda_{\mathcal{R}}(t_f - t_0))w(P)e^{(L+\lambda_R)(t-t_0)}, \tag{4.34}$$

*which is the desired result.* □

**Remark 4** *The assumed first order convergence properties of* $X_0$*,* $S_0$*, and* $\mathcal{R}$ *in* (4.28) *and* (4.29) *are not restrictive. They are satisfied, e.g., if* $X_0$*,* $S_0$*,* $[\mathbf{f}]$*, and* $[\mathbf{f}_s]$ *are natural interval extensions, even if* $\mathcal{R}$ *omits all refinement steps and simply returns* $\Sigma_x = [\mathbf{f}](t, X, P)$ *and* $\Sigma_s = [\mathbf{f}_s](t, X, S, P)$ *[48].*

Next, we show that the mean-value enclosure provided by Theorem 11 has second-order pointwise convergence. This property has been proven previously for DI methods based on more complex sets in [77] and demonstrated empirically but not proven for the DI-based affine bounding method in [19].

**Theorem 14** *Let* $\bar{P} \in \mathbb{IR}^{n_p}$ *and let* $X_0$*,* $S_0$*,* $\mathcal{R}$*, and* $S_P(t)$ *be as in Lemma 3. For every* $P \in \mathbb{I}\bar{P}$*, define the parametric mean-value bounds* $\overline{\mathbf{x}}_P^{MV}, \underline{\mathbf{x}}_P^{MV} : I \times P \to \mathbb{R}^{n_x}$ *by*

$$\begin{aligned} \overline{\mathbf{x}}_P^{MV}(t, \mathbf{p}) &\equiv \max_{\tilde{\mathbf{s}} \in S_P(t)} (\mathbf{x}(t, \hat{\mathbf{p}}) + \tilde{\mathbf{s}}(\mathbf{p} - \hat{\mathbf{p}})), \\ \underline{\mathbf{x}}_P^{MV}(t, \mathbf{p}) &\equiv \min_{\tilde{\mathbf{s}} \in S_P(t)} (\mathbf{x}(t, \hat{\mathbf{p}}) + \tilde{\mathbf{s}}(\mathbf{p} - \hat{\mathbf{p}})). \end{aligned} \tag{4.35}$$

*Then, for every* $t \in I$*,* $\exists \lambda \geq 0$ *such that*

$$\max_{\mathbf{p} \in P} \left\| \overline{\mathbf{x}}_P^{MV}(t, \mathbf{p}) - \underline{\mathbf{x}}_P^{MV}(t, \mathbf{p}) \right\| \leq \lambda w(P)^2, \tag{4.36}$$

*for all* $P \in \mathbb{I}\bar{P}$*.*

**Proof 3** *Choose any $P \in \mathbb{I}\bar{P}$. For any $i \in \{1, \ldots, n_x\}$ and any $(t, \mathbf{p}) \in I \times P$,*

$$\begin{aligned}
&\overline{x}_{P,i}^{MV}(t, \mathbf{p}) - \underline{x}_{P,i}^{MV}(t, \mathbf{p}) \qquad (4.37)\\
&= \max_{\tilde{\mathbf{s}} \in S_P(t)} \sum_j \tilde{s}_{ij}(p_j - \hat{p}_j) - \min_{\tilde{\mathbf{s}} \in S_P(t)} \sum_j \tilde{s}_{ij}(p_j - \hat{p}_j),\\
&= \sum_j \left[ \max_{\tilde{\mathbf{s}} \in S_P(t)} \tilde{s}_{ij}(p_j - \hat{p}_j) - \min_{\tilde{\mathbf{s}} \in S_P(t)} \tilde{s}_{ij}(p_j - \hat{p}_j) \right],\\
&\leq n_p w(S_P(t)) w(P).
\end{aligned}$$

*Choosing any $\lambda$ satisfying Lemma 3, this implies*

$$\max_{\mathbf{p} \in P} \left\| \overline{\mathbf{x}}_P^{MV}(t, \mathbf{p}) - \underline{\mathbf{x}}_P^{MV}(t, \mathbf{p}) \right\| \leq n_p \lambda w(P)^2, \qquad (4.38)$$

*as desired.* □

## 4.7 Numerical Examples

In this section, we compare our new Mean Value Differential Inequalities method (MVDI) to the standard interval-based differential inequalities method (SDI) [64] and other state-of-the-art methods from the literature. MVDI consists of solving the bounding ODEs (4.16) with $\mathcal{R}$ defined by Algorithm 4 with the reference point $\hat{\mathbf{p}} = \text{mid}(P)$. The reference trajectory $\mathbf{x}(t, \hat{\mathbf{p}})$ was obtained by solving (4.1a) with $\mathbf{p} = \hat{\mathbf{p}}$ simultaneously with (4.16). All ODEs were solved using `CVODE` [21] with absolute and relative tolerances of $10^{-6}$ unless explicitly stated otherwise. We report wall clock times for `C++` code running on a laptop with a 2.9 GHz Intel Core i7.

### 4.7.1 Continuous Stirred-Tank Reactor

The first example describes the chemical reactions $A+B \to C$ and $A+C \to D$ in a stirred-tank reactor [19]:

$$\begin{aligned} \dot{A} &= -p_3 AB - k_2 AC + (p_1 v_A - A(\nu_A + \nu_B))/V \\ \dot{B} &= -p_3 AB + (p_2 \nu_B - B(\nu_A + \nu_B))/V \\ \dot{C} &= p_3 AB - k_2 AC - C(\nu_A + \nu_B)/V \\ \dot{D} &= k_2 AC - D(\nu_A + \nu_B)/V \end{aligned} \tag{4.39}$$

The time horizon is $I = [0, 100]$ min and the uncertainties are the inlet concentration of species A, $p_1 \in [0.9, 0.902]$ M, the inlet concentration of species B, $p_2 \in [0.8.802]$ M, and the rate constant of the first reaction, $p_3 \in [10, 10.4]$ $\mathrm{M}^{-1}\mathrm{min}^{-1}$. All other parameters are constant: $V = 20$ L, $k_2 = 0.4$ $\mathrm{M}^{-1}\mathrm{min}^{-1}$, and $\nu_A = \nu_B = 1$ $\mathrm{L(min)}^{-1}$. All concentrations are initially zero.

Figure 4.1 shows the upper and lower bounds on $C$ obtained by MVDI and SDI without considering any invariants or constraints. The figure clearly shows that the use of a mean value enclosure results in bounds that are very accurate and much tighter than SDI. The wall clock time was 0.0013 s for SDI and 0.0350 s for MVDI.

In [19], this example was used to compare a number of alternative bounding methods. There, several methods made use of two invariants that are known to be satisfied by all solutions of (4.39):

$$\begin{aligned} -A + 2B + C &= \gamma(t)(-p_1 \nu_A + 2p_2 \nu_B), \\ A - B + D &= \gamma(t)(p_1 \nu_A - p_2 \nu_B), \end{aligned} \tag{4.40}$$

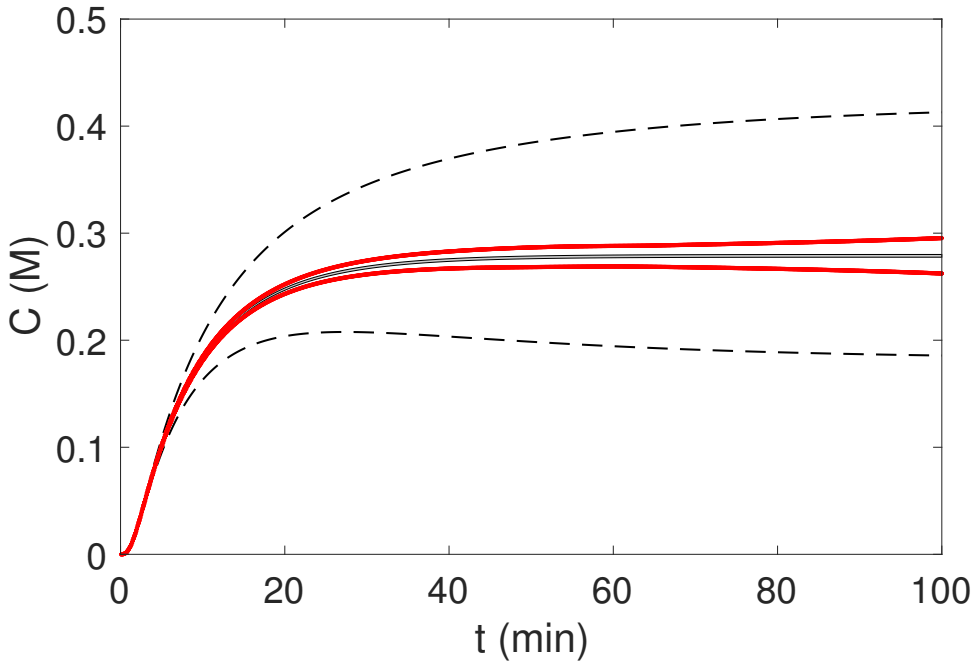


Figure 4.1: Bounds on $C$ in (4.39) computed by standard DI (dashed black) and Mean Value DI without invariants (solid red). Sampled solutions are shaded gray.

where $\gamma(t) = (-\frac{1}{\nu_A+\nu_B})(\exp(-\frac{\nu_A+\nu_B}{V}t) - 1)$. To compare, we also implemented MVDI using these two invariants as $\mathbf{h}^{\text{inv}}$. The most effective methods reported in [19] were the 'Simultaneous Affine/Interval' method, which had a final bound width of $w(X(t_f)) = 0.0131$ and required 0.0296 s, and the 'Simultaneous Affine/Interval (TM)' method, which had $w(X(t_f)) = 0.0120$ and required 0.347 s. In comparison, MVDI achieves the tightest bounds with $w(X(t_f)) = 0.0112$, while also requiring only 0.0257 s.

Figure 4.2 compares the convergence rates of MVDI and SDI by plotting the bounding error versus $w(P)$ on log-log axes. The bound width $w(X(t))$ has first-order convergence (slope 1) for both SDI and MVDI. However, the maximum pointwise error of the mean value enclosure furnished by MVDI, defined as the l.h.s. of (4.36), has second order convergence (slope 2), consistent with Theorem 14.

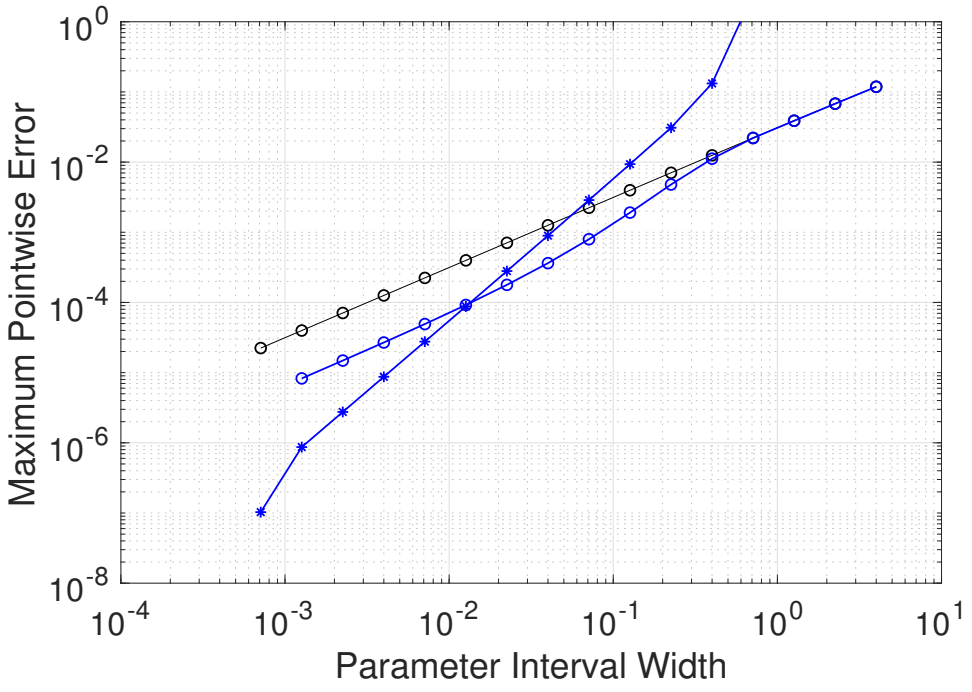


Figure 4.2: Empirical convergence rates for SDI and MVDI (with invariants) applied to (4.39). Black and blue circles show bound width $w(X(t))$ at $t = 100$ min for SDI and MVDI, resp. Blue stars show the maximum pointwise error (l.h.s. of (4.36)) for MVDI.

### 4.7.2 Fixed Wing Unmanned Aerial Vehicle (UAV)

Consider the 7-state fixed-wing UAV model from [1]:

$$\begin{aligned} &\dot{x} = v_{xy}\cos(\psi), \qquad \dot{y} = v_{xy}\sin(\psi), \quad \dot{z} = v_z, \\ &\dot{\psi} = \frac{g}{v_{xy}}\tan(\theta), \quad \dot{v}_{xy} = a_{xy}, \qquad\qquad \dot{v}_z = a_z, \quad \dot{\theta} = \omega. \end{aligned} \tag{4.41}$$

The state of the system is $\mathbf{x} = (x, y, z, \psi, v_{xy}, v_z, \theta)$, where $(x, y, z)$ is the UAV position, $v_{xy}$ and $v_z$ are the velocity in the $xy-$plane and $z-$plane, respectively, $\psi$ is the heading angle, and $\theta$ is the roll angle. The accelerations $a_{xy}$ and $a_z$ and the roll angle rate $\omega$ are control inputs used to track a desired trajectory described by $(x_d, y_d, z_d, v_{xy,d}, v_{z,d}, \psi_d)$, which are all functions of time. The controller equations are

$$\begin{aligned} a_{xy} &= k_5\epsilon_x + k_6(v_{xy,d} - v_{xy}), \\ a_z &= k_7\epsilon_z + k_8(v_{z,d} - v_z), \\ \omega &= k_4(k_1\epsilon_y + k_2(\psi_d - \psi) + k_3(\dot{\psi}_d - \dot{\psi}) - \theta), \\ \epsilon_x &= \cos(\psi_d)(x_d - x) + \sin(\psi_d)(y_d - y), \\ \epsilon_y &= -\sin(\psi_d)(x_d - x) + \cos(\psi_d)(y_d - y), \\ \epsilon_z &= z_d - z. \end{aligned} \tag{4.42}$$

We consider the closed-loop system with time horizon $I = [0, 10]$ s and uncertain initial positions $X_0 = [-1, 1]$ m, $Y_0 = [-1, 1]$ m, and $Z_0 = [-1, 1]$ m. All other initial conditions are assumed fixed at $\psi = 0$ rad, $v_{xy} = 10$ m/s, $v_z = 1$ m/s, and $\theta = 0.3$ rad. Other parameters are chosen as in [1]: $k_1 = 0.05$, $k_2 = 5.0$, $k_3 = 5.0$, $k_4 = 1.0$, $k_5 = 0.1$, $k_6 = 1.0$, $k_7 = 0.13$, and $k_8 = 1.0$. The desired trajectory is obtained by solving (4.41) with initial conditions $x_d = 0$ m, $y_d = 0$ m, $z_d = 0$ m, $\psi_d = 0$

rad, $v_{xy,d} = 10$ m/s, and $v_{z,d} = 1$ m/s, and with the open-loop inputs $\theta_d = 0.3$ rad, $a_{xy,d} = 1$ m/s, and $a_{z,d} = 0.1$ m/s. This system has no known invariants and we do not impose any constraints.

Figure 4.3 shows that SDI produces rapidly diverging bounds, while MVDI produces very sharp bounds. The time required for integrating a single trajectory of (4.41) is $1.4 \times 10^{-4}$s on average, while SDI takes $5.8 \times 10^{-3}$s and MVDI takes $1.6 \times 10^{-2}$s. Thus, MVDI produces accurate bounds over 10s of flight time more than three orders of magnitude faster than real-time. Moreover, computing a rigorous enclosure by MVDI is less costly than simulating trajectories on a $4 \times 4 \times 4$ grid over the uncertain initial condition space (64 trajectories at a total cost of $9.0 \times 10^{-3}$s), which is unlikely to provide an reliable approximation of the full reachable set.

## 4.8 Appendix

Theorem 11 and 12 are proven here as special cases of a more general bounding theorem that is not in principle related to sensitivities and the mean value theorem. It is convenient to state this result with notation that is distinct from that in §4.2. Therefore, let $I = [t_0, t_f]$, let $Q \subset \mathbb{R}^{n_q}$ be a compact set of time-invariant parameters $\mathbf{q}$, let $\mathbf{d} : D_d \subset \mathbb{R} \times \mathbb{R}^{n_y} \times \mathbb{R}^{n_q} \to \mathbb{R}^{n_y}$ and $\mathbf{y}_0 \colon Q \to \mathbb{R}^{n_y}$ be locally Lipschitz continuous functions, and consider the system

$$\dot{\mathbf{y}}(t, \mathbf{q}) = \mathbf{d}(t, \mathbf{y}(t, \mathbf{q}), \mathbf{q}), \quad \mathbf{y}(t_0, \mathbf{q}) = \mathbf{y}_0(\mathbf{q}). \tag{4.43}$$

We assume that (4.43) admits a unique continuous solution $\mathbf{y} : I \times Q \to \mathbb{R}^{n_y}$ such that $\mathbf{y}(\cdot, \mathbf{q})$ is absolutely continuous on $I$ for every $\mathbf{q} \in Q$.

Our general bounding result is stated in terms of interval operators $\Pi_i^L, \Pi_i^U$ :

$E_\Pi \subset I \times \mathbb{IR}^{n_y} \times \mathbb{IR}^{n_y} \to \mathbb{IR}$. These functions are generic notation for any operations that take intervals $Y$ and $\tilde{Y}$ as input, isolate the $i^{\text{th}}$ lower or upper face of $Y$, and then eliminate regions of this face that violate some known constraints satisfied by the solutions of (4.43). Eventually, these will be used to represent the refinement operations used in Theorems 11 and 12. The arguments $Y$ and $\tilde{Y}$ are related to the discussion surrounding $S$ and $\tilde{S}$ in §4.3. Specifically, these are necessary to allow $\Pi_i^{L/U}$ to make refinements that are only valid under the assumption $Y$ contains $\mathbf{y}(t, \mathbf{q})$ for some specific $\mathbf{q} \in Q$, while $\tilde{Y}$ contains $\mathbf{y}(t, \mathbf{q})$ for all $\mathbf{q} \in Q$. The general requirements for $\Pi_i^{L/U}$ are given in the following assumption.

**Assumption 8** *For every $i \in \{1, \cdots, n_y\}$, assume that*

1. *If $(t, Y, \tilde{Y}) \in E_\Pi$ and $\bar{\mathbf{q}} \in Q$ satisfy $\mathbf{y}(t, \bar{\mathbf{q}}) \in \mathcal{B}_i^L(Y)$ and $\mathbf{y}(t, \mathbf{q}) \in \tilde{Y}$, $\forall \mathbf{q} \in Q$, then $\dot{y}_i(t, \bar{\mathbf{q}}) \in \Pi_i^L(t, Y, \tilde{Y})$.*

2. *If $(t, Y, \tilde{Y}) \in E_\Pi$ and $\bar{\mathbf{q}} \in Q$ satisfy $\mathbf{y}(t, \bar{\mathbf{q}}) \in \mathcal{B}_i^U(Y)$ and $\mathbf{y}(t, \mathbf{q}) \in \tilde{Y}$, $\forall \mathbf{q} \in Q$, then $\dot{y}_i(t, \bar{\mathbf{q}}) \in \Pi_i^U(t, Y, \tilde{Y})$.*

3. *$E_\Pi$ is an open with respect to $I \times \mathbb{IR}^{n_y} \times \mathbb{IR}^{n_y}$. That is, for any $(t, Y, \tilde{Y}) \in E_\Pi$, $\exists \eta > 0$ such that $B_\eta((t, Y, \tilde{Y})) \cap (I \times \mathbb{IR}^{n_y} \times \mathbb{IR}^{n_y})$ is a subset of $E_\Pi$.*

4. *$\Pi_i^L$ and $\Pi_i^U$ are locally Lipschitz continuous.*

Theorem 15 and Corollary 3 below are proven under Assumption 8 in [68].

**Theorem 15** *Let $\mathbf{y}^L, \mathbf{y}^U : I \to \mathbb{R}^{n_y}$ be absolutely continuous and denote $Y(t) \equiv [\mathbf{y}^L(t), \mathbf{y}^U(t)]$. Assume that:*

1. *$(t, Y(t), Y(t)) \in E_\Pi$, $\forall t \in I$.*

2. *$\mathbf{y}_0(\mathbf{q}) \in Y(t_0)$, $\forall \mathbf{q} \in Q$.*

*3. For almost every $t \in I$ and each index $i$,*

*(a)* $\dot{y}_i^L \leq \sigma$, $\forall \sigma \in \Pi_i^L(t, Y(t), Y(t))$,

*(b)* $\dot{y}_i^U \geq \sigma$, $\forall \sigma \in \Pi_i^U(t, Y(t), Y(t))$.

*Then* $\mathbf{y}(t, \mathbf{q}) \in Y(t)$, $\forall (t, \mathbf{q}) \in I \times Q$.

**Remark 5** *Note that Assumption 8 treats the second and third arguments of* $\Pi_i^{L/U}$ *differently, while Theorem 15 supplies* $Y(t)$ *in both positions. This is intentional and is necessitated by technical details of the proof in [68]. Suffice it to say here that* $\Pi_i^{L/U}$ *is evaluated with several other arguments in the course of the proof.*

**Corollary 3** *Let* $Y_0 : Q \to \mathbb{IR}^{n_y}$ *be an inclusion function for* $\mathbf{y}_0$ *and let* $\mathbf{y}^L, \mathbf{y}^U : I \to \mathbb{R}^{n_y}$ *satisfy the following system of ODEs with* $Y(t) \equiv [\mathbf{y}^L(t), \mathbf{y}^U(t)]$*:*

$$\begin{aligned} \dot{y}_i^L(t) &= \min\{\sigma_i : \sigma_i \in \Pi_i^L(t, Y(t), Y(t))\}, \\ \dot{y}_i^U(t) &= \max\{\sigma_i : \sigma_i \in \Pi_i^U(t, Y(t), Y(t))\}, \\ Y(t_0) &= Y_0(Q). \end{aligned} \tag{4.44}$$

*Then* $\mathbf{y}(t, \mathbf{q}) \in Y(t)$, $\forall (t, \mathbf{q}) \in I \times Q$.

We now prove Theorem 11 as a direct application of Corollary 3 to the system (4.8) with $Q = P$, $\mathbf{q} = \mathbf{p}$, $\mathbf{y}(t, \mathbf{q}) = \begin{bmatrix} \mathbf{x}(t,\mathbf{p}) \\ \mathbf{s}(t,\mathbf{p}) \end{bmatrix}$, and $\mathbf{d}(t, \mathbf{y}, \mathbf{q}) = \begin{bmatrix} \mathbf{f}(t,\mathbf{x},\mathbf{p}) \\ \mathbf{f}_s(t,\mathbf{x},\mathbf{s},\mathbf{p}) \end{bmatrix}$. Let $\mathcal{R}$ satisfy Definition 14, let $X(t)$ and $S(t)$ satisfy the hypotheses of Theorem 11, and define $Y(t) \equiv \begin{bmatrix} X(t) \\ S(t) \end{bmatrix}$. Similarly, define $Y_0 \equiv \begin{bmatrix} X_0 \\ S_0 \end{bmatrix}$ and, for arbitrary $(\tilde{X}, \tilde{S}) \in \mathbb{IR}^{n_x} \times \mathbb{IR}^{n_x \times n_p}$, define $\tilde{Y} \equiv \begin{bmatrix} \tilde{X} \\ \tilde{S} \end{bmatrix}$. For any $Z \in \mathbb{IR}^n$, define the coordinate projection $\pi_i(Z) \equiv Z_i$. Define $\Pi_i^{L/U}$ as follows, where $i \in \{1, \ldots, n_x\}$, $j \in \{1, \ldots, n_p\}$, and $k = n_x + n_x(j -$

$1) + i$:

$$E_\Pi \equiv \left\{ (t, Y, \tilde{Y}): \begin{array}{l} t \in I \\ (t, P, \mathcal{B}_i^{L/U}(X), S, \tilde{S}) \in D_\mathcal{R}, \ \forall i \\ (t, P, X, \mathcal{B}_{ij}^{L/U}(S), \tilde{S}) \in D_\mathcal{R}, \ \forall i, j \end{array} \right\},$$

$$\Pi_i^L(t, Y, \tilde{Y}) \equiv \pi_i \circ \mathcal{R}(t, P, \mathcal{B}_i^L(X), S, \tilde{S}),$$

$$\Pi_i^U(t, Y, \tilde{Y}) \equiv \pi_i \circ \mathcal{R}(t, P, \mathcal{B}_i^U(X), S, \tilde{S}),$$

$$\Pi_k^L(t, Y, \tilde{Y}) \equiv \pi_k \circ \mathcal{R}(t, P, X, \mathcal{B}_{ij}^L(S), \tilde{S}),$$

$$\Pi_k^U(t, Y, \tilde{Y}) \equiv \pi_k \circ \mathcal{R}(t, P, X, \mathcal{B}_{ij}^U(S), \tilde{S}). \tag{4.45}$$

With these definitions, the bounding ODEs in Corollary 3 are equivalent to those in Theorem 11. Thus, it only remains to show that (4.45) satisfies Assumption 8.

**Lemma 4** *If $\mathcal{R}$ satisfies Definition 14, then the definitions* (4.45) *satisfy Assumption 8.*

**Proof 4** *To verify Condition 1 of Assumption 8, choose any $i \in \{1, \ldots, n_y\}$, any $(t, Y, \tilde{Y}) \in E_\Pi$, and any $\bar{\mathbf{q}} \in Q$ satisfying $\mathbf{y}(t, \bar{\mathbf{q}}) \in \mathcal{B}_i^L(Y)$ and $\mathbf{y}(t, \mathbf{q}) \in \tilde{Y}$, $\forall \mathbf{q} \in Q$. Assume first that $i \leq n_x$. In this case, $\dot{y}_i(t, \bar{\mathbf{q}}) = \dot{x}_i(t, \bar{\mathbf{p}}) = f_i(t, \mathbf{x}(t, \bar{\mathbf{p}}), \bar{\mathbf{p}})$ and $\Pi_i^L(t, Y, \tilde{Y}) \equiv \pi_i \circ \mathcal{R}(t, P, \mathcal{B}_i^L(X), S, \tilde{S})$. Thus, we must prove that $f_i(t, \mathbf{x}(t, \bar{\mathbf{p}}), \bar{\mathbf{p}}) \in \pi_i \circ \mathcal{R}(t, P, \mathcal{B}_i^L(X), S, \tilde{S})$. It suffices to prove that $(\mathbf{f}(t, \mathbf{x}(t, \bar{\mathbf{p}}), \bar{\mathbf{p}}), \mathbf{f}_s(t, \mathbf{x}(t, \bar{\mathbf{p}}), \mathbf{s}(t, \bar{\mathbf{p}}), \bar{\mathbf{p}}))$ is an element of $\mathcal{R}(t, P, \mathcal{B}_i^L(X), S, \tilde{S})$ by the definition of $\pi_i$. This follows from Condition 1 of Definition 14. Specifically, the hypothesis $\mathbf{y}(t, \bar{\mathbf{q}}) \in \mathcal{B}_i^L(Y)$ implies that $\mathbf{x}(t, \bar{\mathbf{p}}) \in \mathcal{B}_i^L(X)$ and $\mathbf{s}(t, \bar{\mathbf{p}}) \in S$. Moreover, the hypothesis $\mathbf{y}(t, \mathbf{q}) \in \tilde{Y}$, $\forall \mathbf{q} \in Q$, implies that $\mathbf{s}(t, \mathbf{p}) \in \tilde{S}$, $\forall \mathbf{p} \in P$. By Corollary 2, it follows that $\exists \tilde{\mathbf{s}} \in \tilde{S}$ satisfying $\mathbf{x}(t, \bar{\mathbf{p}}) = \mathbf{x}(t, \hat{\mathbf{p}}) + \tilde{\mathbf{s}}(\bar{\mathbf{p}} - \hat{\mathbf{p}})$. Moreover,* (4.4) *and* (4.12) *imply that $\mathbf{g}^{inv}(t, \mathbf{x}(t, \bar{\mathbf{p}}), \bar{\mathbf{p}}) \leq \mathbf{0}$,*

$\mathbf{h}^{inv}(t, \mathbf{x}(t, \bar{\mathbf{p}}), \bar{\mathbf{p}}) = \mathbf{0}$, *and*

$$\frac{\partial \mathbf{h}^{\text{inv}}}{\partial \mathbf{x}}(t, \mathbf{x}(t, \bar{\mathbf{p}}), \bar{\mathbf{p}})\mathbf{s}(t, \bar{\mathbf{p}}) + \frac{\partial \mathbf{h}^{\text{inv}}}{\partial \mathbf{p}}(t, \mathbf{x}(t, \bar{\mathbf{p}}), \bar{\mathbf{p}}) = \mathbf{0}.$$

*Therefore, the point* $(\bar{\mathbf{p}}, \mathbf{x}(t, \bar{\mathbf{p}}), \mathbf{s}(t, \bar{\mathbf{p}}), \tilde{\mathbf{s}})$ *satisfies all of the constraints in Condition 1 of Definition 14, and hence* $(\mathbf{f}(t, \mathbf{x}(t, \bar{\mathbf{p}}), \bar{\mathbf{p}}), \mathbf{f}_s(t, \mathbf{x}(t, \bar{\mathbf{p}}), \mathbf{s}(t, \bar{\mathbf{p}}), \bar{\mathbf{p}}))$ *is an element of* $\mathcal{R}(t, P, \mathcal{B}_i^L(X), S, \tilde{S})$. *This verifies Condition 1 of Assumption 8 for* $i \leq n_x$. *The proof for* $i > n_x$, *as well as the proof of Condition 2 of Assumption 8, both follow from very similar arguments and are omitted for brevity.*

*To verify Condition 3 of Assumption 8, note that the mappings* $(t, Y, \tilde{Y}) \mapsto (t, P, \mathcal{B}_i^{L/U}(X), S, \tilde{S})$ *and* $(t, Y, \tilde{Y}) \mapsto (t, P, X, \mathcal{B}_{ij}^{L/U}(S), \tilde{S})$ *are continuous on the metric space* $I \times \mathbb{IR}^{n_y} \times \mathbb{IR}^{n_y}$ *(see §4.1.1). Moreover,* $D_{\mathcal{R}}$ *is open by Condition 3 of Definition 14. Thus, the preimage of* $D_{\mathcal{R}}$ *under each of these mappings is open with respect to* $I \times \mathbb{IR}^{n_y} \times \mathbb{IR}^{n_y}$. *Since* $E_\Pi$ *is simply the intersection of these preimages, it is also open with respect to* $I \times \mathbb{IR}^{n_y} \times \mathbb{IR}^{n_y}$.

*To verify Condition 4 of Assumption 8, note that the interval functions* $\mathcal{B}_i^{L/U}$ *and* $\pi_i$ *are Lipschitz continuous on* $\mathbb{IR}^n$ *(see §4.1.1). Moreover,* $\mathcal{R}$ *is locally Lipschitz continuous by Condition 2 of Definition 14. Thus, each* $\Pi_i^{L/U}$ *is a composition of locally Lipschitz continuous functions, and is therefore locally Lipschitz continuous.*

□

In light of Lemma 4, applying Corollary 3 with the definitions above ensures that $\mathbf{x}(t, \mathbf{p}) \in X(t)$ and $\mathbf{s}(t, \mathbf{p}) \in S(t)$, $\forall (t, \mathbf{p}) \in I \times P$. It then follows from Corollary 2 that $\mathbf{x}(t, \mathbf{p}) \in \mathbf{x}(t, \hat{\mathbf{p}}) + S(t)(\mathbf{p} - \hat{\mathbf{p}})$, $\forall (t, \mathbf{p}) \in I \times P$. This completes the proof of Theorem 11.

Next, Theorem 12 is proven by applying Corollary 3 to (4.1a)–(4.1b) with $Q = P^*$, $\mathbf{q} = \mathbf{p}$, $\mathbf{y}(t, \mathbf{q}) = \mathbf{x}(t, \mathbf{p})$, and $\mathbf{d}(t, \mathbf{y}, \mathbf{q}) = \mathbf{f}(t, \mathbf{x}, \mathbf{p})$. Let $\mathcal{R}^*$ satisfy Definition

16 and let $S(t)$ and $X^*(t)$ satisfy the hypotheses of Theorem 12. Define $Y(t) \equiv X^*(t)$, $Y_0 \equiv X_0$, and for arbitrary $X \in \mathbb{IR}^{n_x}$, $Y \equiv X$. Define $\Pi_i^{L/U}$ as follows, where $i \in \{1, \ldots, n_x\}$:

$$E_\Pi \equiv \left\{ (t, Y, \tilde{Y}) : \begin{array}{l} t \in I \\ (t, P, \mathcal{B}_i^{L/U}(X), S(t)) \in D_\mathcal{R}, \ \forall i \end{array} \right\},$$

$$\Pi_i^L(t, Y, \tilde{Y}) \equiv \pi_i \circ \mathcal{R}^*(t, P, \mathcal{B}_i^L(X), S(t)),$$

$$\Pi_i^U(t, Y, \tilde{Y}) \equiv \pi_i \circ \mathcal{R}^*(t, P, \mathcal{B}_i^U(X), S(t)). \tag{4.46}$$

With these definitions, the bounding ODEs in Corollary 3 are equivalent to those in Theorem 12. Thus, it only remains to show that (4.46) satisfies Assumption 8.

**Lemma 5** *If $\mathcal{R}^*$ satisfies Definition 16, then the definitions* (4.46) *satisfy Assumption 8.*

**Proof 5** *To verify Condition 1 of Assumption 8, choose any $i \in \{1, \ldots, n_y\}$, any $(t, Y, \tilde{Y}) \in E_\Pi$, and any $\bar{\mathbf{q}} \in Q$ satisfying $\mathbf{y}(t, \bar{\mathbf{q}}) \in \mathcal{B}_i^L(Y)$ and $\mathbf{y}(t, \mathbf{q}) \in \tilde{Y}$, $\forall \mathbf{q} \in Q$. These conditions imply that $\bar{\mathbf{p}} \in P^*$ and $\mathbf{x}(t, \bar{\mathbf{p}}) \in \mathcal{B}_i^L(X)$. Since $\dot{y}_i(t, \bar{\mathbf{q}}) = \dot{x}_i(t, \bar{\mathbf{p}}) = f_i(t, \mathbf{x}(t, \bar{\mathbf{p}}), \bar{\mathbf{p}})$ and $\Pi_i^L(t, Y, \tilde{Y}) \equiv \pi_i \circ \mathcal{R}^*(t, P, \mathcal{B}_i^L(X), S(t))$, we must prove that $f_i(t, \mathbf{x}(t, \bar{\mathbf{p}}), \bar{\mathbf{p}}) \in \pi_i \circ \mathcal{R}^*(t, P, \mathcal{B}_i^L(X), S(t))$. By the definition of $\pi_i$, it suffices to prove that $\mathbf{f}(t, \mathbf{x}(t, \bar{\mathbf{p}}), \bar{\mathbf{p}})$ is an element of $\mathcal{R}^*(t, P, \mathcal{B}_i^L(X), S(t))$. Since $\mathbf{s}(t, \mathbf{p}) \in S(t)$, $\forall \mathbf{p} \in P$, Corollary 2 ensures that $\exists \tilde{\mathbf{s}} \in S(t)$ satisfying $\mathbf{x}(t, \bar{\mathbf{p}}) = \mathbf{x}(t, \hat{\mathbf{p}}) + \tilde{\mathbf{s}}(\bar{\mathbf{p}} - \hat{\mathbf{p}})$. Moreover, since $\bar{\mathbf{p}} \in P^*$, we have $\mathbf{g}(t, \mathbf{x}(t, \bar{\mathbf{p}}), \bar{\mathbf{p}}) \leq \mathbf{0}$ and $\mathbf{h}(t, \mathbf{x}(t, \bar{\mathbf{p}}), \bar{\mathbf{p}}) = \mathbf{0}$. Therefore, the point $(\bar{\mathbf{p}}, \mathbf{x}(t, \bar{\mathbf{p}}), \tilde{\mathbf{s}})$ satisfies all of the constraints in Condition 1 of Definition 16, and hence $\mathbf{f}(t, \mathbf{x}(t, \bar{\mathbf{p}}), \bar{\mathbf{p}})$ is an element of $\mathcal{R}^*(t, P, \mathcal{B}_i^L(X), S(t))$. This verifies Condition 1 of Assumption 8, and Condition 2 follows analogously.*

*To verify Condition 3 of Assumption 8, note that the mappings $(t, Y, \tilde{Y}) \mapsto$*

*$(t, P, \mathcal{B}_i^{L/U}(X), S(t))$ are continuous on the metric space $I \times \mathbb{IR}^{n_y} \times \mathbb{IR}^{n_y}$ (see §4.1.1). Moreover, $D_{\mathcal{R}^*}$ is open by Condition 3 of Definition 14. Thus, the preimage of $D_{\mathcal{R}^*}$ under each of these mappings is open with respect to $I \times \mathbb{IR}^{n_y} \times \mathbb{IR}^{n_y}$. Since $E_\Pi$ is simply the intersection of these preimages, it is also open with respect to $I \times \mathbb{IR}^{n_y} \times \mathbb{IR}^{n_y}$.*

*To verify Condition 4 of Assumption 8, note that the functions $\mathcal{B}_i^{L/U}$, $\pi_i$, and $S$ are locally Lipschitz continuous. Moreover, $\mathcal{R}^*$ is locally Lipschitz continuous by Condition 2 of Definition 16. Thus, each $\Pi_i^{L/U}$ is a composition of locally Lipschitz continuous functions, and is therefore locally Lipschitz continuous.* □

In light of Lemma 5, applying Corollary 3 with the definitions above ensures that $\mathbf{x}(t, \mathbf{p}) \in X^*(t)$, $\forall (t, \mathbf{p}) \in I \times P^*$. This completes the proof of Theorem 12.

### 4.8.1 Modifications to Algorithm 4

Algorithm 4 provides a general way to refine an interval $X \times P \times S \times \tilde{S}$ based on mean value relations and pre-existing invariants. However, considering Algorithm 4, it can be identified that all major refinement computation is done inside the inner loop, i.e., lines 5–15. For large systems, the refinement can be expensive. In this section, a simplified algorithm with computational analysis is provided. Throughout this section, we assume that no constraints and invariants exist, which is more general although the resulting simplified algorithm is still valid in case of other existing constraints/invariants. We first provide a new simplified Algorithm 5, which we use in all numerical experiments, then a computational analysis follows.

As it shows, the major difference between Algorithm 4 and 5 is that Algorithm 5 first figures out which right-hand-side of the ODEs is currently being integrated by checking its index variable, namely, the argument $i$ of the refinement operator $\mathcal{R}$

**Algorithm 5** A Simplified implementation of $\mathcal{R}$

1: **function** $\mathcal{R}(t, P, X, S, \tilde{S}, i)$
2: **if** $i \leq n_x$ **then**
3: **for** $j \leftarrow 1$ to $n_p$ **do**
4: $\alpha \leftarrow \hat{x}_i - X_i + \sum_{k \neq j} \tilde{S}_{ik}(P_k - \hat{p}_k)$
5: $\mu \leftarrow 1/\max(\epsilon, |\tilde{S}_{ij}|)$
6: $P'_j \leftarrow \mu\alpha + (1 + \mu\tilde{S}_{ij})(P_j - \hat{p}_j) + \hat{p}_j$
7: $P_j \leftarrow P_j \bar{\cap} P'_j$
8: $\mu \leftarrow -\mu$ and repeat lines 6-7
9: $\mu \leftarrow 1/\max(\epsilon, |P_j - \hat{p}_j|)$
10: $\tilde{S}'_{ij} \leftarrow \mu\alpha + (1 + \mu(P_j - \hat{p}_j))\tilde{S}_{ij}$
11: $\tilde{S}_{ij} \leftarrow \tilde{S}_{ij} \bar{\cap} \tilde{S}'_{ij}$
12: $\mu \leftarrow -\mu$ and repeat lines 10-11
13: **end for** ▷ $j \leftarrow 1$ to $n_p$
14: **for** $j \leftarrow 1$ to $i-1, i+1$ to $n_x$ **do**
15: $X'_j \leftarrow \hat{x}_j + \sum_{k=1}^{n_p} \tilde{S}_{jk}(P_k - \hat{p}_k)$
16: $X_j \leftarrow X_j \bar{\cap} X'_j$
17: **for** $k \leftarrow 1, n_p$ **do**
18: repeat lines 9–12
19: **end for** ▷ $k \leftarrow 1$ to $n_p$
20: **end for** ▷ $j \leftarrow 1$ to $i-1, i+1$ to $n_x$
21: $P^\dagger \leftarrow P, X^\dagger \leftarrow X, \tilde{S}^\dagger \leftarrow \tilde{S}$
22: $\Sigma_x \leftarrow [\mathbf{f}](t, X^\dagger, P^\dagger) \cap F_{MV}(t, P^\dagger, X^\dagger, \tilde{S}^\dagger)$
23: **else**
24: **for** $j \leftarrow 1$ to $n_x$ **do**
25: $X'_j \leftarrow \hat{x}_j + \sum_{k=1}^{n_p} \tilde{S}_{jk}(P_k - \hat{p}_k)$
26: $X_j \leftarrow X_j \bar{\cap} X'_j$
27: **end for** ▷ $i \leftarrow 1$ to $n_x$
28: $P^\dagger \leftarrow P, X^\dagger \leftarrow X, S^\dagger \leftarrow S$
29: $\Sigma \leftarrow [\mathbf{f}_s](t, X^\dagger, S^\dagger, P^\dagger)$
30: **end if**
31: **return** $\Sigma$
32: **end function**

defined in Algorithm 5. Remember that an essential feature of using (4.10) is that it enables to refine $P$. Furthermore, the capability of refining $P$ by (4.10) is only possible when $X_i$ is flattened. Therefore, it's more effective to refine $P$ and $S_i$ first as lines 3–13. Next, for non flattened intervals $X_{j\neq i}$, it just needs to refine $X_{j\neq i}$ and $\tilde{S}_j$ as lines 14–20. Furthermore, for the r.h.s. of $\mathbf{f}_s$, since all interval variables are just state bounds, $P$ cannot be refined. In addition, there is also no need to refine $\tilde{S}$ since they are not used in $[\mathbf{f}_s]$.

**Remark 6** *Algorithm 5 along above tricks explained are what actually used in all numerical experiments. Besides that, it is worth to mention that there are several tricks which are not implemented in Algorithm 5 (and numerical experiments) but could be combined with Algorithm 5 to reduce the computational cost.*

*First, for each r.h.s. of $s_{ik}$, since all intervals of the r.h.s. in line 25 are actually state bounds (no flatten), rather than computing lines 24–27 $n_x n_p$ times for each $s_{ik}$ with $i \in \{1, \cdots, n_x\}$ and $k \in \{1, \cdots, n_p\}$, they could be computed just once and used by all $s_{ij}$.*

*Second, considering line 6, it can be easily verified that if $0 < [P_j - \hat{p}_j]^L$ and $0 \in \alpha$, we have $P'_j \supset P_j$, therefore lines 6–7 can be skipped without doing redundant computation. Anagolusly, simplification can be made for the case of $\mu = -\mu$ and the refinement of $\tilde{S}_{ij}$.*

*Third, for dynamic systems, since the refinement operator is implemented for every $t \in I$, a variable occurrence matrix can be provided such that only those interval variables appearing on the corresponding r.h.s. will be refined. This is especially useful when evaluating lines 14–20.*

*Finally, inspired by the variable occurrence, as we show in* (4.25)*, by expanding $\mathbf{p}$ first, we have $\hat{\mathbf{p}}$ as the argument of $\frac{\partial f_i}{\partial x_k}(t, \boldsymbol{\gamma}_{\not i}, x_i, \hat{\mathbf{p}})$ which would be $\frac{\partial f_i}{\partial x_k}(t, \boldsymbol{\gamma}_{\not i}, \gamma_i, \boldsymbol{\zeta})$*

*if we expand both* $\mathbf{p}$ *and* $\mathbf{x}$ *simultaneously. This partially expanding has obvious advantage when evaluating the natural interval extension of* (4.25). *That is, instead of only using interval as the argument, partially replacing interval with corresponding reference number (which can be considered as interval whose lower and upper bound is same) could provide much tighter enclosures, based on the so-called 'inclusion property' in [48]. Following this strategy, one could expand* $f_i$ *with respect to the variable/parameter first which has most occurrence, then gradually to the variable/parameters which has the least occurrence. This will enable to use more reference points in place of unknown components in* $\boldsymbol{\zeta}$ *and* $\boldsymbol{\gamma}$*, which we could call it as fully component-wise* $F_{MV}$*.*

# Chapter 5

# Conclusion and Future Work

## 5.1 Conclusion

This dissertation considers the problem of computing rigorous, fast, and accurate enclosures of the solutions of nonlinear ordinary differential equations subject to bounded initial conditions, parameters and time-varying inputs. Our aim is to produce such enclosures at the cost of a few single simulation, and with much higher accuracy than existing methods of similar complexity. In particular, this dissertation continues the development of differential inequalities methods [64] that exploit redundant model equations to reduce bounding conservatism. Our aim is achieved from two directions. First, a new framework is developed for introducing redundant model equations, called 'manufactured invariants', for general nonlinear systems. Introducing such manufactured invariants lifts the original governing ODEs into a higher-dimensional state-space, often resulting in significantly improved enclosures. Problem specific user insights are needed in order to manually identify effective manufactured invariants. To address this, a new approach that can automate the construction of manufactured invariants based on the forward sensitivity system for the given ODEs

is developed. This new approach is very competitive and extends effective DI methods base on invariants to many dynamic systems where pre-existing redundant model equations are not available and manually derived manufactured invariants are not easy to create. This solves the 'how to create redundant model equations' problem. Second, fast and accurate new bounding algorithms and their underlying theories that exploit linear and nonlinear redundant model equations to significantly reduce conservatism are developed. This solves the 'how to optimally exploit redundant model equations' problem.

Combining contributions from these two directions, many numerical examples demonstrate that these new developed differential inequalities-based methods can dramatically reduce conservatism while maintaining high efficiency much better than many other existing methods. As a result, the newly proposed methods can potentially be used as a potential tool for many on-line applications that require both fast and accurate enclosures of the system states.

## 5.2 Future work

Our key insight is that the conservatism of fast interval methods can be dramatically reduced through the use of redundant model equations. As a result, there are a few areas that may deserve future research efforts.

First, efficient algorithms have been developed in Chapters 2, 3, and 4 for exploiting linear and nonlinear redundant model equations. However, improvements could be made in many aspects. Algorithms that can exploit the most effective redundant model equations first rather than using all of them in a fixed sequence should be focused. This is necessary because large system may involve hundreds of redundant model equations or constraints, and having a smart selective algorithm

can save significant computational resources. For large system with dense linear redundant model equations, more effective algorithms should be developed.

Second, the ability to refine enclosures continuously on the right-hand-sides of the ODEs make the new methods in this dissertation very effective. Actually continuous refinement is expensive. It is what makes the bounds tight, but not what makes them efficient. However, the Lipschitz continuity requirement for the right-hand-sides of the ODEs also brings some limitations. At least for current algorithms that exploit nonlinear redundant model equations, conservatism is actually introduced in order to satisfy the Lipschitz continuity requirement. More efficient and effective refinement algorithms for exploiting nonlinear redundant model equations should be made in the future.

Chapter 4 introduces one way to automate the introduction of redundant model equations by augmenting the original system with its forward sensitivity equations. In fact, this is also one way to mitigate the dependency problem between states and parameters. However, what should be noticed is that, for some systems, the enclosures of the forward sensitivities often diverge earlier and more quickly than the original states due to the *dependency problem*. In these cases, the refinement based on the mean value relations that is introduced in Chapter 4 may not be as effective as it is in other cases. Therefore, a direction for future research is establishing other automated ways to manufacture redundant model equations.

Interval-based differential inequalities methods have a significant efficiency advantage over many other existing methods that use more complex sets. However, considering the stability of bounds, interval-based DI methods still need to be improved. More research that tries to combine interval-based DI methods with other modern bounding methods should be made. However, hard work needs to be done to find the balance point such that these 'hybrid' methods can still provide fast and

accurate bounds for real-time applications.

Finally, the proposed methods have been shown to be promising for providing fast and accurate bounds over a single range of uncertain parameters, which is often called parent node. For global optimization problems of dynamic system, the parent node may be partitioned to thousands of child nodes. In principle, the proposed new methods are much more efficient than the old bounding methods. However, in the context of global optimization, since relaxation methods are often used, interplay with relaxations and the proposed methods should be made such that they can benefit from each other. For systems with constraints, advanced methods that can exploit constraints and the mean value enclosures in Chapter 4 to conduct domain reduction are also should be focused.

# Bibliography


[1] D. Althoff, M. Althoff, and S. Scherer. Online safety verification of trajectories for unmanned flight with offline computed robust invariant sets. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015.

[2] M. Althoff, M. Cvetkovic, and M. Ilic. Transient stability analysis by reachable set computation. In *2012 3rd IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)*, pages 1–8, 2012.

[3] M. Althoff and J. M. Dolan. Online verification of automated road vehicles using reachability analysis. *IEEE Transactions on Robotics*, 30(4):903–918, Aug 2014.

[4] M. Althoff and B.H. Krogh. Reachability analysis of nonlinear differential-algebraic systems. *IEEE T. Automat. Contr.*, 59(2):371–383, 2014.

[5] M. Althoff, O. Stursberg, and M. Buss. Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization. In *47th IEEE Decis. Contr. P.*, pages 4042–4048, 2008.

[6] Olivier Bernard, Zakaria Hadj-Sadok, Denis Dochain, Antoine Genovesi, and Jean-Philippe Steyer. Dynamical model development and parameter identification for an anaerobic wastewater treatment process. *Biotechnol. Bioeng.*, 75(4):424–438, 2001.

[7] Martin Berz and Kyoko Makino. Performance of Taylor model methods for validated integration of ODEs. In *Proceedings of the 7th International Conference on Applied Parallel Computing: State of the Art in Scientific Computing*, pages 65–73. Springer-Verlag, 2006.

[8] Davide Bresolin, Luca Geretti, Riccardo Muradore, Paolo Fiorini, and Tiziano Villa. Formal verification of robotic surgery tasks by reachability analysis. *Microprocessors and Microsystems*, 39(8):836–842, 2015.

[9] Benoit Chachuat and Mario Villanueva. Bounding the solutions of parametric ODEs: When Taylor models meet differential inequalities. In I. D. L. Bogle and M. Fairweather, editors, *22 European Symposium on Computer Aided Process*

*Engineering*, volume 30 of *Comput. Aided Chem. Eng.*, pages 1307–1311. Elsevier Science BV, 2012.

[10] Earl A. Coddington and Norman Levinson. *Theory of Ordinary Differential Equations*. McGraw-Hill, New York, 1955.

[11] Evrim Dalkiran and Hanif D. Sherali. Theoretical filtering of RLT bound-factor constraints for solving polynomial programming problems to global optimality. *J. Global Optim.*, 57(4):1147–1172, 2013.

[12] Iman Famili and Bernhard O. Palsson. The convex basis of the left null space of the stoichiometric matrix leads to the definition of metabolically meaningful pools. *Biophys. J.*, 85:16–26, 2003.

[13] H.P. Geering. *Optimal Control with Engineering Applications*. Springer Berlin Heidelberg, 2007.

[14] H. Gueguen, M. A. Lefebvre, J. Zaytoon, and O. Nasri. Safety verification and reachability analysis for hybrid systems. *Annual Reviews in Control*, 33(1):25–36, 2009.

[15] K. Hariprasad and S. Bhartiya. Adaptive robust model predictive control of nonlinear systems using tubes based on interval inclusions. In *53rd IEEE Decis. Contr. P.*, pages 2032–2037, 2014.

[16] G. W. Harrison. Dynamic models with uncertain parameters. In X.J.R. Avula, editor, *Proc. of the First International Conference on Mathematical Modeling*, volume 1, pages 295–304, 1977.

[17] S. M. Harwood and P. I. Barton. Efficient polyhedral enclosures for the reachable set of nonlinear control systems. *Math. Control. Signal.*, 28(1):8, 2016.

[18] S. M. Harwood, J. K. Scott, and P. I. Barton. Bounds on reachable sets using ordinary differential equations with linear programs embedded. *IMA J. Math. Control I.*, 33(2):519–541, 2016.

[19] Stuart M. Harwood and Paul I. Barton. Affine relaxations for the solutions of constrained parametric ordinary differential equations. *Optim. Contr. Appl. Met.*, pages 1–22, 2017.

[20] Stuart M. Harwood and Paul I. Barton. How to solve a design centering problem. *Math. Meth. of OR*, 86:215–254, 2017.

[21] A. C. Hindmarsh, P. N. Brown, K. E. Grant, S. L. Lee, R. Serban, D. E. Shumaker, and C. S. Woodward. SUNDIALS, suite of nonlinear and differential/algebraic equation solvers. *ACM T. Math. Software*, 31:363–396, 2005.

[22] B. Houska and B. Chachuat. Branch-and-Lift algorithm for deterministic global optimization in nonlinear optimal control. *J. Optim. Theor. Appl.*, 162(1):208–248, 2014.

[23] B. Houska, M. E. Villanueva, and B. Chachuat. Stable set-valued integration of nonlinear dynamic systems using affine set-parameterizations. *SIAM J. Numer. Anal.*, 53(5):2307–2328, 2015.

[24] B. Houska, M.E. Villanueva, and B. Chachuat. A validated integration algorithm for nonlinear ODEs using Taylor models and ellipsoidal calculus. In *52nd IEEE Decis. Contr. P.*, pages 484–489, 2013.

[25] John Ingham, Irving J Dunn, Elmar Heinzle, Jirí E Prenosil, and Jonathan B Snape. *Chemical engineering dynamics: An introduction to modelling and computer simulation*, volume 3. John Wiley & Sons, 2008.

[26] A. Agung Julius and George J. Pappas. *Trajectory Based Verification Using Local Finite-Time Invariance*, pages 223–236. Springer, Berlin, Heidelberg, 2009.

[27] Aida Khajavirad, JeremyJ. Michalek, and NikolaosV. Sahinidis. Relaxations of factorable functions with convex-transformable intermediates. *Mathematical Programming*, 144(1–2):107–140, 2014.

[28] M. Kishida and R. D. Braatz. Skewed structured singular value-based approach for the construction of design spaces: Theory and applications. *IET Control Theory A.*, 8(14):1321–1327, 2014.

[29] Masako Kishida and Richard D. Braatz. A model-based approach for the construction of design spaces in quality-by-design. *2012 American Control Conference (ACC)*, pages 1513–1518, 2012.

[30] A.B. Kurzhanski. Hamiltonian techniques for the problem of set-membership state estimation. *Int. J. Adapt. Control Signal Process.*, 25(3):249–263, 2011.

[31] Jean-Paul P. Laumond. *Robot Motion Planning and Control*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1998.

[32] D. Limon, J. M. Bravo, T. Alamo, and E. F. Camacho. Robust MPC of constrained nonlinear systems based on interval arithmetic. *IEE P-Contr. Theor. Ap.*, 152(3):325–332, 2005.

[33] Youdong Lin and Mark A. Stadtherr. Deterministic global optimization of nonlinear dynamic systems. *AIChE J.*, 53(4):866–875, 2007.

[34] Youdong Lin and Mark A. Stadtherr. Validated solutions of initial value problems for parametric ODEs. *Applied Numerical Mathematics*, 57(10):1145–1162, 2007.

[35] Youdong Lin and Mark A. Stadtherr. Fault detection in nonlinear continuous-time systems with uncertain parameters. *AIChE J.*, 54(9):2335–2345, 2008.

[36] Youdong Lin and Mark A. Stadtherr. Rigorous model-based safety analysis for nonlinear continuous-time systems. *Comput. Chem. Eng.*, 33(2):493 – 502, 2009.

[37] J. Maidens and M. Arcak. Reachability analysis of nonlinear systems using matrix measures. *IEEE Trans. Automat. Contr.*, 60(1):265–270, 2015.

[38] K. Makino and M. Berz. Efficient control of the dependency problem based on Taylor model methods. *Reliable Computing*, 5:3–12, 1999.

[39] Kyoko Makino and Martin Berz. Taylor models and other validated functional inclusion methods. *Int. J. Pure Appl. Math*, 4(4):379–456, 2003.

[40] Ruth Misener and Christodoulos A. Floudas. ANTIGONE: Algorithms for continuous/integer global optimization of nonlinear equations. *J. Global Optim.*, 59(2-3):503–526, 2014.

[41] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin. A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games. *IEEE Trans. Automat. Contr.*, 50(7):947–957, July 2005.

[42] M. Moisan, O. Bernard, and J. L. Gouze. Near optimal interval observers bundle for uncertain bioreactors. *Automatica*, 45(1):291–295, 2009.

[43] R. E. Moore. *Methods and Applications of Interval Analysis*. SIAM, Philadelphia, PA, 1979.

[44] Ramon E. Moore. *Interval Analysis*. Prentice-Hall, Englewood Cliffs, N.J., 1966.

[45] James R. Munkres. *Analysis on Manifolds*. Westview Press, Cambridge, MA, 1991.

[46] Z. K. Nagy and R. D. Braatz. Open-loop and closed-loop robust optimal control of batch processes using distributional and worst-case analysis. *J. Process Contr.*, 14(4):411–422, 2004.

[47] N. S. Nedialkov, K. R. Jackson, and G. F. Corliss. Validated solutions of initial value problems for ordinary differential equations. *Applied Mathematics and Computation*, 105(1):21–68, 1999.

[48] A. Neumaier. *Interval Methods for Systems of Equations*. Cambridge University Press, Cambridge, 1990.

[49] C. Perez-Galvan and I. D. L. Bogle. *Deterministic Global Dynamic Optimisation using Interval Analysis*, volume 37 of *Computer Aided Chemical Engineering*, pages 791–796. 2015.

[50] N. Peric, M. Villanueva, and B. Chachuat. Sensitivity analysis of uncertain dynamic systems using set-valued integration. *SIAM J. Sci. Comput.*, 2017.

[51] H. N. V. Pico and D. C. Aliprantis. Voltage ride-through capability verification of wind turbines with fully-rated converters using reachability analysis. *IEEE T. Energy. Conver.*, 29(2):392–405, 2014.

[52] A. U. Raghunathan, V. Gopal, D. Subramanian, L. T. Biegler, and T. Samad. Dynamic optimization strategies for three-dimensional conflict resolution of multiple aircraft. *J. Guid. Control. Dynam.*, 27(4):586–594, 2004.

[53] D.M. Raimondo, G.R. Marseglia, R.D. Braatz, and J.K. Scott. Closed-loop input design for guaranteed fault diagnosis using set-valued observers. *Automatica*, 74:107–117, 2016.

[54] T. Raissi, N. Ramdani, and Y. Candau. Set membership state and parameter estimation for systems described by nonlinear differential equations. *Automatica*, 40(10):1771–1777, 2004.

[55] F. Rossi, S. Copelli, A. Colombo, C. Pirola, and F. Manenti. Online model-based optimization and control for the combined optimal operation and runaway prediction and prevention in (fed-)batch systems. *Chem. Eng. Sci.*, 138:760–771, 2015.

[56] Spencer D. Schaber, Joseph K. Scott, and Paul I. Barton. Convergence-order analysis for differential-inequalities-based bounds and relaxations of the solutions of ODEs. *Journal of Global Optimization*, 2018.

[57] J. K. Scott. *Reachability Analysis and Deterministic Global Optimization of Differential-Algebraic Systems*. PhD thesis, Massachusetts Institute of Technology, April 2012.

[58] J. K. Scott and P. I. Barton. Tight, efficient bounds on the solutions of chemical kinetics models. *Comput. Chem. Eng.*, 34:717–731, 2010.

[59] J. K. Scott and P. I. Barton. Interval Bounds on the Solutions of Semi-Explicit Index-One DAEs. Part 1: Analysis. *Numer. Math.*, 125(1):1–25, 2011.

[60] J. K. Scott, D. M. Raimondo, G. R. Marseglia, and R. D. Braatz. Constrained zonotopes: A new tool for set-based estimation and fault detection. *Automatica*, 69:126–136, 2016.

[61] J.K. Scott and P.I. Barton. Interval Bounds on the Solutions of Semi-Explicit Index-One DAEs. Part 2: Computation. *Numer. Math.*, 125(1):27–60, 2011.

[62] J.K. Scott and P.I. Barton. Reachability analysis and deterministic global optimization of DAE models. In Achim Ilchman and Timo Reis, editors, *Surveys in Differential Algebraic Equations III*, volume 3, pages 61–116. Springer International Publishing, 2015.

[63] Joseph K. Scott and Paul I Barton. Convex enclosures for the reachable sets of nonlinear parametric ordinary differential equations. In *49th IEEE Decis. Contr. P.*, pages 5695–5700, 2010.

[64] Joseph K. Scott and Paul I. Barton. Bounds on the reachable sets of nonlinear control systems. *Automatica*, 49(1):93–100, 2013.

[65] Joseph K. Scott and Paul I. Barton. Improved relaxations for the parametric solutions of ODEs using differential inequalities. *J. Global Optim.*, 57(1):143–176, 2013.

[66] Kai Shen and Joseph K. Scott. Exploiting nonlinear invariants and path constraints to achieve tighter bounds on the flows of uncertain nonlinear systems using differential inequalities. *Mathematics of Control, Signals, and Systems.* submitted.

[67] Kai Shen and Joseph K. Scott. Rapid and accurate reachability analysis for nonlinear dynamic systems by exploiting model redundancy. *Comput. Chem. Eng.*, 106:596 – 608, 2017. SI: ESCAPE-26.

[68] Kai Shen and Joseph K. Scott. Mean value form enclosures for nonlinear reachability analysis. In *Proc. of 57th IEEE Conference on Decision and Control (CDC)*, 2018.

[69] Kai Shen and Joseph K. Scott. Tight reachability bounds for nonlinear systems using nonlinear and uncertain solution invariants. In *Proc. of the American Control Conference*, 2018.

[70] H. D. Sherali and W. P. Adams. A reformulation-linearization technique (RLT) for semi-infinite and convex programs under mixed 0-1 and general discrete restrictions. *Discrete Appl. Math.*, 157(6):1319–1333, 2009.

[71] T.C. Sideris. *Ordinary Differential Equations and Dynamical Systems.* Atlantis Studies in Differential Equations. Atlantis Press, 2013.

[72] A. B. Singer and P. I. Barton. Bounding the solutions of parameter dependent nonlinear ordinary differential equations. *SIAM J. Sci. Comput.*, 27(6):2167–2182, 2006.

[73] A. B. Singer, J. W. Taylor, P. I. Barton, and W. H. Green. Global dynamic optimization for parameter estimation in chemical kinetics. *J. Phys. Chem. A.*, 110(3):971–976, 2006.

[74] A.B. Singer and P.I. Barton. Global optimization with nonlinear ordinary differential equations. *J. Glob. Optim.*, 34:159–190, 2006.

[75] B. Srinivasan, D. Bonvin, E. Visser, and S. Palanki. Dynamic optimization of batch processes ii. role of measurements in handling uncertainty. *Comput. Chem. Eng.*, 27(5):761–761, 2003.

[76] M. E. Villanueva, B. Houska, and B. Chachuat. On the stability of set-valued integration for parametric nonlinear ODEs. *24 European Symposium on Computer Aided Process Engineering*, 33:595–600, 2014.

[77] M. E. Villanueva, B. Houska, and B. Chachuat. Unified framework for the propagation of continuous-time enclosures for parametric nonlinear ODEs. *J. Global Optim.*, 62(3):575–613, 2015.

[78] Mario E. Villanueva, Rien Quirynen, Moritz Diehl, Benoit Chachuat, and Boris Houska. Robust MPC via minmax differential inequalities. *Automatica*, 77:311–321, 2017.

[79] Xuan-Ha Vu, Hermann Schichl, and Djamila Sam-Haroud. Interval propagation and search on directed acyclic graphs for numerical constraint solving. *J. Global Optim.*, 45(4):499–531, 2009.

[80] Y. Wang, D. M. Bevly, and R. Rajamani. Interval observer design for LPV systems with parametric uncertainty. *Automatica*, 60:79–85, 2015.

[81] Achim Wechsung, Spencer D. Schaber, and Paul I. Barton. The cluster problem revisited. *J. Global Optim.*, 58(3):429–438, 2014.