AFRL-AFOSR-VA-TR-2019-0248

RAPID AND ACCURATE UNCERTAINTY PROPAGATION FOR NONLINEAR DYNAMIC SYSTEMS BY EXPLOITING MODEL REDUNDANCY

Joseph Scott
CLEMSON UNIVERSITY

07/29/2019
Final Report

Air Force Research Laboratory
AF Office Of Scientific Research (AFOSR)/ RTA2
Arlington, Virginia 22203
Air Force Materiel Command

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
20-08-2019	Final Performance	01 May 2016 to 30 Apr 2019

4. TITLE AND SUBTITLE
RAPID AND ACCURATE UNCERTAINTY PROPAGATION FOR NONLINEAR DYNAMIC SYSTEMS BY EXPLOITING MODEL REDUNDANCY

5a. CONTRACT NUMBER

5b. GRANT NUMBER
FA9550-16-1-0158

5c. PROGRAM ELEMENT NUMBER
61102F

6. AUTHOR(S)
Joseph Scott

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
CLEMSON UNIVERSITY
201 SIKES HALL
CLEMSON, SC 29634-0001 US

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AF Office of Scientific Research
875 N. Randolph St. Room 3112
Arlington, VA 22203

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/AFOSR RTA2

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AFRL-AFOSR-VA-TR-2019-0248

12. DISTRIBUTION/AVAILABILITY STATEMENT
A DISTRIBUTION UNLIMITED: PB Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The objective of this project was to develop new methods for rapidly computing accurate bounds on the solutions of nonlinear ordinary differential equations (ODEs) subject to bounded uncertainties. Toward this end, our key insight was that the conservatism of fast interval methods can be dramatically reduced through the use of model redundancy. Specifically, prior work showed that bounds produced by interval methods often enclose large regions of state-space that violate redundant relations implied by the dynamics, such as conservation laws. Furthermore, such relations (known as 'solution invariants') can be exploited to obtain much sharper bounds in many cases. Motivated by these observations, we pursued new bounding approaches for general nonlinear systems based on the deliberate introduction of redundant model equations to reduce conservatism. The work was organized around three major tasks. The most significant accomplishments in each task are summarized below:
Task 1: Develop a fast and accurate state bounding algorithm that exploits pre-existing model redundancy. A new bounding theorem was proven that enables the use of nonlinear invariants within fast bounding methods based on differential inequalities (DI) for the first time. An efficient new bounding algorithm was also developed to implement this theory. In aggregate, these advances extend the redundancy-based DI bounding approach to systems satisfying a much more general class of invariants and have enabled efficient computation of very sharp bounds for several test cases.
Task 2: Develop a theoretical framework for the introduction of redundancy into arbitrary dynamic models to effectively reduce conservatism. A framework was developed for introducing solution invariants into arbitrary systems by lifting them into a higher-dimensional state space. Critically, this enables the methods from Task 1 to be appli

15. SUBJECT TERMS
uncertainty propagation, reachability set, interval arithmetic

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			LEVE, FREDERICK
Unclassified	Unclassified	Unclassified	UU		19b. TELEPHONE NUMBER *(Include area code)* 703-696-7309

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

AFOSR Final Progress Report

Project Title: Rapid and Accurate Uncertainty Propagation for Nonlinear Dynamic Systems by Exploiting Model Redundancy

Award Number: FAA9550-16-1-0158

Start Date: 1 May 2016

Reporting Period: 1 May 2016 – 30 April 2019

Program Officer: Dr. Frederick A. Leve
Dynamics and Control
Air Force Office of Scientific Research
Directorate of Information and Networks
frederick.leve@us.af.mil

Principal Investigator: Prof. Joseph K. Scott
Department of Chemical and Biomolecular Engineering
Clemson University
jks9@clemson.edu
(864) 656-0997

Contents

1 Executive Summary

The objective of this project was to develop new methods for rapidly computing accurate bounds on the solutions of nonlinear ordinary differential equations (ODEs) subject to bounded uncertainties. It has long been possible to compute such bounds efficiently using interval methods, but the results are often too conservative to be useful. In contrast, some modern strategies can achieve remarkably sharp bounds, but are too costly for real-time decision making. Our aim here was to produce bounds at a small multiple of the cost of simulating a single trajectory but with much higher accuracy than existing methods of similar complexity.

Toward this end, our key insight was that the conservatism of fast interval methods can be dramatically reduced using model redundancy. Specifically, our prior work showed that bounds produced by interval methods often enclose large regions of state-space that violate redundant relations implied by the dynamics, such as conservation laws. Furthermore, such relations (known as *solution invariants*) can be exploited to obtain much sharper bounds in many cases. Motivated by these observations, we pursued new bounding approaches for general nonlinear systems based on the deliberate introduction of redundant model equations to reduce conservatism. The work was organized around three major tasks. The most significant accomplishments in each task are summarized below:

Task 1: *Develop a fast and accurate state bounding algorithm that exploits pre-existing model redundancy.* A new bounding theorem was proven that enables the use of nonlinear invariants within fast bounding methods based on differential inequalities (DI) for the first time [1,2]. An efficient new bounding algorithm was also developed to implement this theory. In aggregate, these advances extend the redundancy-based DI bounding approach to systems satisfying a much more general class of invariants and have enabled efficient computation of very sharp bounds for several test cases.

Task 2: *Develop a theoretical framework for the introduction of redundancy into arbitrary dynamic models to effectively reduce conservatism.* A framework was developed for introducing solution invariants into arbitrary systems by lifting them into a higher-dimensional state space [3]. Critically, this enables the methods from Task 1 to be applied to systems that do not initially satisfy any invariants. However, the user must specify effective invariants to introduce, which often requires considerable insight. Thus, tailored strategies were developed for (i) transient mass and energy balance models [3] and (ii) vehicle models under path and trajectory tracking control [4]. Moreover, a new method called mean-value differential inequalities (MVDI) was developed that automatically constructs effective invariants for arbitrary systems using the forward sensitivity system [5,6]. Numerous test cases show that these methods can provide sharp bounds at very low cost.

Task 3: *Develop algorithms and software for fast and accurate state bounding through the automatic identification, introduction, and exploitation of model redundancy.* MVDI is the most effective method discovered in this project for automatically identifying, introducing, and exploiting invariants. An efficient implementation of MVDI was developed in `C++` and is described in [6].

Through interactions enabled by this project, we became aware of several closely related problems of interest to AFOSR that were not included in the original scope. In two cases, we were able to make significant progress by direct extensions of the methods above. First, we developed new methods for bounding the solutions of discrete-time systems rather than ODEs [7,8]. Many robust estimation and control problems of interest are commonly formulated in discrete-time. Thus, valid discrete-time analogous of the advanced DI methods above were developed and shown to offer significant advantages over existing methods for some test cases. Second, a new method was developed for solving a class of formal verification problems that can be formulated as backward reachability problems for nonlinear ODEs [9]. Such problems can be solved by embedding many forward reachability calculations in a branch-and-bound framework. Preliminary results herein show that using the advanced DI methods discussed above for these forward calculations enables the solution of some verification problems much more efficiently than the current state of the art.

2 Research Findings

2.1 Background Relevant to the Findings Described in this Report

This project considered dynamic systems described by nonlinear ODEs of the form

$$\dot{x}(t) = f(t,x(t),w(t)), \quad x(t_0) = x_0, \quad t \in [t_0,t_f], \quad x(t) \in \mathbb{R}^{n_x}, \quad w(t) \in \mathbb{R}^{n_w}, \tag{1}$$

with $x_0 \in X_0$ and measurable disturbances $w : [t_0,t_f] \to \mathbb{R}^{n_x}$ satisfying $w(t) \in W$ for interval uncertainty sets $X_0 = [x_0^L, x_0^U]$ and $W = [w^L, w^U]$. The objective was to compute an accurate enclosure of the *reachable set*

$$\mathscr{R}(t) \equiv \{x(t;x_0,w) : x_0 \in X_0,\ w(s) \in W,\ \forall s \in [t_0,t_f]\}. \tag{2}$$

The methods investigated are based on the theory of differential inequalities (DI) [10], which states that two functions $x^L, x^U : [t_0,t_f] \to \mathbb{R}^{n_x}$ are guaranteed to satisfy $\mathscr{R}(t) \subset X(t) \equiv [x^L(t), x^U(t)]$, $\forall t \in [t_0,t_f]$, provided that they satisfy the following differential inequalities for all $i = 1,\ldots,n_x$:

$$\dot{x}_i^L(t) \le \min\{f_i(t,z,v) : v \in W,\ z \in [x^L(t),x^U(t)],\ z_i = x_i^L(t)\}, \quad x_i^L(t_0) \le x_{0,i}^U, \tag{3}$$

$$\dot{x}_i^U(t) \ge \max\{f_i(t,z,v) : v \in W,\ z \in [x^L(t),x^U(t)],\ z_i = x_i^U(t)\}, \quad x_i^U(t_0) \ge x_{0,i}^U. \tag{4}$$

The DI approach computes bounds by first constructing an auxiliary system of ODEs that describes x^L and x^U satisfying (3)–(4) as its solution, and then solving this system numerically [10]. To make this precise, let $\beta_i^L(x^L(t),x^U(t)) \equiv \{z \in [x^L(t),x^U(t)] : z_i = x_i^L(t)\}$ and $\beta_i^U(x^L(t),x^U(t)) \equiv \{z \in [x^L(t),x^U(t)] : z_i = x_i^U(t)\}$. These sets are intervals and represent exactly the set of z's feasible in the optimizations in (3) and (4), respectively. Then, bounds are computed as the solutions of the following system of $2n_x$ ODEs:

$$\dot{x}_i^L(t) = f_i^L(t,\beta_i^L(x^L(t),x^U(t)),W), \quad x_i^L(t_0) = x_{0,i}^U, \tag{5}$$

$$\dot{x}_i^U(t) = f_i^U(t,\beta_i^U(x^L(t),x^U(t)),W), \quad x_i^U(t_0) = x_{0,i}^U, \tag{6}$$

where f_i^L and f_i^U are lower and upper bounds on the range of f_i over the specified ranges of arguments computed using interval arithmetic. Using a state-of-the-art numerical integrator, bounds can be obtained from (5)–(6) at a small multiple of the cost of integrating a single trajectory of (1), making DI a potentially powerful tool for real-time applications [11]. However, this basic approach produces extremely conservative bounds in general.

Prior to this project, the PI developed a method for computing sharper DI-based bounds for systems that satisfy affine solution invariants [11]. A *solution invariant* is a function $g : \mathbb{R}^{n_x} \to \mathbb{R}^{n_m}$ such that $g(x(t;x_0,w)) = 0$ for all $t \in [t_0,t_f]$ and all $(x_0,w) \in X_0 \times W$, and is affine if $g(z) \equiv Mz - b = 0$ for some $M \in \mathbb{R}^{n_m \times n_x}$ and $b \in \mathbb{R}^{n_m}$. For such systems, improved bounds are given as the solutions of

$$\dot{x}_i^L(t) = f_i^L(t,\mathscr{I}_g[\beta_i^L(x^L(t),x^U(t))],W), \quad x_i^L(t_0) = x_{0,i}^U, \tag{7}$$

$$\dot{x}_i^U(t) = f_i^U(t,\mathscr{I}_g[\beta_i^U(x^L(t),x^U(t))],W), \quad x_i^U(t_0) = x_{0,i}^U, \tag{8}$$

where $\mathscr{I}_g$ is an interval refinement operator that must satisfy $\mathscr{I}_g(Z) \supset \{z \in Z : Mz = b\}$, $\forall Z \in \mathbb{IR}^{n_x}$, as well as the following regularity condition, where d_H denotes the Hausdorff metric:

$$\exists L \in \mathbb{R}_+ : \quad d_H(\mathscr{I}_g(Z_1),\mathscr{I}_g(Z_2)) \le L d_H(Z_1,Z_2), \quad \forall Z_1,Z_2 \in \mathbb{IR}^{n_x}. \tag{9}$$

The use of the refinement operator $\mathscr{I}_g$ has been shown to dramatically improve the accuracy of the bounds computed through (7)–(8) relative to standard DI for many case studies. Moreover, the additional computational cost is typically very modest. However, this method only applies to models that naturally satisfy affine solution invariants (e.g., conservation laws), and therefore offers no useful solution for the majority of systems of practical interest. The purpose of this project was to pursue a novel extension of this approach to general nonlinear systems through the deliberate introduction of redundant model equations and invariants.

2.2 Research Findings from Task 1

The objective of Task 1 was to develop a fast differential inequalities (DI) bounding approach that produces much tighter bounds than existing methods by exploiting model redundancy in the form of (potentially nonlinear) solution invariants. Related research prior to this project applied only to affine invariants [11]. Thus, new theory and methods were necessary to handle nonlinear invariants. Moreover, the affine method in [11] was not optimized for either efficiency or accuracy. The results of this project on both of these issues are described in the subsections below. These results are discussed in further detail in the publications [1–3].

2.2.1 Optimal use of Affine Invariants through New Preconditioning Techniques

A new technique was developed for preconditioning systems of affine invariants that leads to significant improvements in the accuracy the bounds computed by the DI method in [11]. To clarify this contribution, consider a system (1) that is known to satisfy affine invariants $Mx(t;x_0,w) = b$ for all $t \in [t_0,t_f]$ and all $(x_0,w) \in X_0 \times W$. Moreover, consider computing bounds $X(t) \equiv [x^L(t),x^U(t)]$ by solving the ODEs (7)–(8) with $\mathscr{I}_g$ satisfying $\mathscr{I}_g(Z) \supset \{z \in Z : Mz = b\}$, $\forall Z \in \mathbb{IR}^{n_x}$, and the Lipschitz condition (9). The article [11] proposes an algorithm for $\mathscr{I}_g$ that produces an interval enclosure of $\{z \in Z : Mz = b\}$ efficiently using iterative interval computations. However, this method considers the equations in the system $Mz = b$ one at a time, so the resulting interval $\mathscr{I}_g(Z)$ is highly sensitive to M and b. Specifically, a different enclosure $\mathscr{I}_g(Z)$ is obtained if $Mz = b$ is first preconditioned by an invertible matrix P; i.e., $PMz = Pb$.

Based on this observation, we investigated preconditioning methods that would result in the sharpest enclosure $\mathscr{I}_g(Z)$ for a given Z. The resulting theory is described in [3]. The key points are as follows:

- For a given interval Z and a given $i \in \{1,\ldots,n_x\}$, there exists a vector $\mu \in \mathbb{R}^{n_m}$ such that the invariant $\mu^{\mathrm{T}}Mz = \mu^{\mathrm{T}}b$ is optimal for refining the lower bound z_i^L using the $\mathscr{I}_g$ operator previously described in [11], and a distinct vector $\gamma \in \mathbb{R}^{n_m}$ such that $\gamma^{\mathrm{T}}Mz = \gamma^{\mathrm{T}}b$ is optimal for refining the upper bound z_i^U. Moreover, these vectors can be obtained for each i as the solutions of $2n$ simple linear programs.

- The optimal preconditioning vectors described above depend on Z. However, in the course of integrating the bounding ODEs (7)–(8), $\mathscr{I}_g$ is applied to many intervals that are not known in advance. Since solving $2n_x$ linear programs at every time step during integration is impractical, we attempted to characterize preconditioning vectors that are robust in the sense that they are provably optimal for an entire range of intervals Z. We found that optimal vectors exist that remain optimal under (i) translation of Z along the the null space of M, and (ii) scaling of Z with fixed relative edge lengths. Thus, our new method computes $2n_x$ preconditioning vectors at t_0 based on a representative interval enclosure (possibly physically motivated), which are then guaranteed to be optimal for a wide range of interval arguments (although certainly not all) that may be encountered during the integration of (7)–(8).

Example 1. The following nonlinear ODEs describe the time-evolution of the concentrations of six chemical species involved in an enzymatic reaction network:

$$\begin{aligned}
\dot{x}_{\mathrm{A}} &= -k_1 x_{\mathrm{A}} x_{\mathrm{F}} + k_2 x_{\mathrm{F:A}} + k_6 x_{\mathrm{R:A'}} \\
\dot{x}_{\mathrm{F}} &= -k_1 x_{\mathrm{A}} x_{\mathrm{F}} + k_2 x_{\mathrm{F:A}} + k_3 x_{\mathrm{F:A}} \\
\dot{x}_{\mathrm{F:A}} &= k_1 x_{\mathrm{A}} x_{\mathrm{F}} - k_2 x_{\mathrm{F:A}} - k_3 x_{\mathrm{F:A}} \\
\dot{x}_{\mathrm{A'}} &= k_3 x_{\mathrm{F:A}} - k_4 x_{\mathrm{A'}} x_{\mathrm{R}} + k_5 x_{\mathrm{R:A'}} \\
\dot{x}_{\mathrm{R}} &= -k_4 x_{\mathrm{A'}} x_{\mathrm{R}} + k_5 x_{\mathrm{R:A'}} + k_6 x_{\mathrm{R:A'}} \\
\dot{x}_{\mathrm{R:A'}} &= k_4 x_{\mathrm{A'}} x_{\mathrm{R}} - k_5 x_{\mathrm{R:A'}} - k_6 x_{\mathrm{R:A'}}
\end{aligned} \tag{10}$$

Let $[t_0,t_f] = [0,0.04]$ s, $x_0 = (34,20,0,0,16,0)$ M, and let the rate parameters $k = (k_1,\ldots,k_6)$ be uncertain and lie within the admissible set $W = [\hat{k},10\hat{k}]$ with $\hat{k} = (0.1,0.033,16,5,0.5,0.3)$.

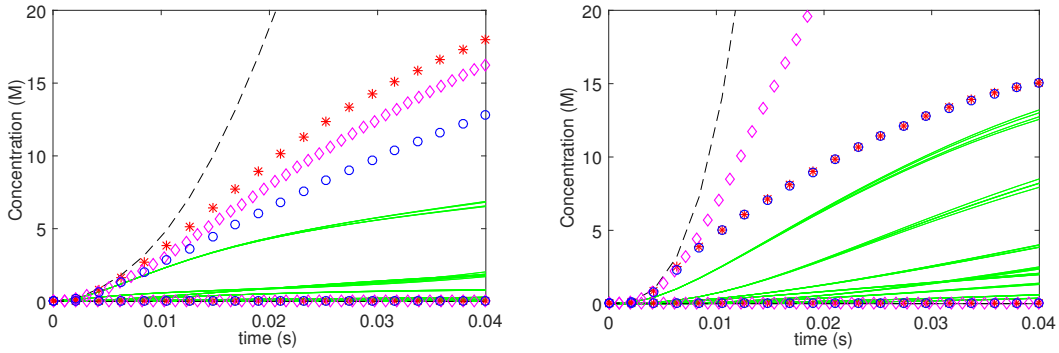

Figure 1: State bounds for $x_{A'}$ (left) and $x_{RA'}$ (right) from Example 1, computed using standard DI (dashed) and by solving (7)–(8) using affine invariants $Mx(t) = Mx_0$ with M derived from MATLAB's `null` (red star), M as in (11) (magenta diamond), and M derived by our new optimal preconditioning algorithm (blue circles).

Dynamic models for chemical reaction systems of this type can be written in the form $\dot{x}(t) = Sr(t,x(t),k)$, where r is a vector of nonlinear reaction rate functions and $S \in \mathbb{R}^{n_x \times n_r}$ is the stoichiometry matrix, which encodes the topology of the reaction network. It is well known that such systems satisfy the affine solution invariants $Mx(t;x_0,k) = Mx_0$ for any M whose rows lie in the left null space of S. Of course, this choice is not unique. Figure 11 shows state bounds for (10) computed by solving the bounding system (7)–(8) using $\mathscr{I}_g$ as defined in [11] with three different choices of M. The first has rows that form an orthonormal basis for the left null space of S obtained using the MATLAB subroutine `null`. The second is given by

$$M = \begin{bmatrix} 0 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & -1 \\ 1 & -1 & 0 & 1 & -1 & 0 \end{bmatrix}. \tag{11}$$

This choice is physically motivated by conservation laws and is regarded as the best matrix that one could reasonably determine based on sound physical understanding of the system. Finally, the third choice of M is obtained by starting from (11) and applying the new preconditioning method described above.

Fig. 1 show that our new preconditioning algorithm leads to substantially sharper bounds. In particular, it avoids the large upper-bounding error observed in the right panel of Fig. 1 when using (11).

2.2.2 A New Algorithm for Exploiting Affine Invariants with Reduced Computational Complexity

A new implementation of the refinement operator $\mathscr{I}_g$ originally defined in [11] was developed that has lower computational complexity by a factor of n_x (the number of state variables). Since this operator is called in every time-step during the integration of the bounding ODEs (7)–(8), this results in significant speed-ups.

Recall that $\mathscr{I}_g$ refines a given interval Z based on affine invariants $Mz = b$ and must satisfy $\mathscr{I}_g(Z) \supset \{z \in Z : Mz = b\}$, $\forall Z \in \mathbb{IR}^{n_x}$. To achieve such a refinement, the original idea from [11] is to consider, for each $m_{ij} \neq 0$, the rearrangement of the i^{th} equation for z_j,

$$z_j = m_{ij}^{-1}(b_i - \textstyle\sum_{k \neq j} m_{ik} z_k). \tag{12}$$

Given an initial interval $Z = [z^L, z^U]$, the j^{th} component $[z_j^L, z_j^U]$ can potentially be refined by bounding the right-hand side of (12) using interval arithmetic. For example, (12) implies that

$$\frac{b_i}{m_{ij}} + \sum_{k \neq j} \min(-\frac{m_{ik}}{m_{ij}} z_k^L, -\frac{m_{ik}}{m_{ij}} z_k^U) \leq z_j \leq \frac{b_i}{m_{ij}} + \sum_{k \neq j} \max(-\frac{m_{ik}}{m_{ij}} z_k^L, -\frac{m_{ik}}{m_{ij}} z_k^U), \tag{13}$$

which potentially provides improved bounds $[z_j^L, z_j^U]$ at a cost of $O(n_x)$ flops. In [11], $\mathscr{I}_g$ is defined by applying such refinements to every possible choice of $m_{ij} \neq 0$, of which there are at most $n_m n_x$ for $M \in \mathbb{R}^{n_m \times n_x}$, for a total complexity of $O(n_m n_x^2)$.

In our new implementation of $\mathscr{I}_g$, we reduce this to $O(n_m n_x)$ by eliminating repeated computations. Specifically, for a fixed equation $m_i^{\mathrm{T}} z = b_i$, the original implementation computes the sums in (13) independently for every j. However, 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 first computing the sums $b_i + \sum_{k=1}^{n_x} \min(-m_{ik} z_k^L, -m_{ik} z_k^U)$ and $b_i + \sum_{k=1}^{n_x} \max(-m_{ik} z_k^L, -m_{ik} z_k^U)$. This has complexity $O(n_x)$, but is done only once. Then, the sum on the right-hand side of (13) can be computed for each j by a simple $O(1)$ update. With this change, the complexity of valuating $\mathscr{I}_g$ is reduced by a factor of n_x, which in turn reduces the complexity of evaluating the right-hand side of the bounding ODE system (7)–(8) by a total of $2n_x^2$ flops.

2.2.3 A New Differential Inequalities Bounding Theorem Enabling the Use of Nonlinear Invariants

A new bounding theorem was formulated and proven that enables the use of nonlinear invariants within differential inequalities methods for the first time. This theorem extends the central result of [11], which shows that if the solutions of (1) satisfy an invariant of the form $g(x(t;x_0,w)) = 0$, then a valid interval enclosure of these solutions is provided by the solutions of (7)–(8), where $\mathscr{I}_g$ is an interval refinement operator that must satisfy $\mathscr{I}_g(Z) \supset \{z \in Z : g(z) = 0\}$ as well as the Lipschitz condition (9). Strictly speaking, this prior result applies to nonlinear invariants, but a specific definition of $\mathscr{I}_g$ satisfying the required properties when g is nonlinear was not known prior to this project. More importantly, it become clear in the course of this project that the result in [11] relies on some simplifying assumptions that all but exclude the use of nonlinear invariants in practice. Thus, a major achievement of this project was to formulate and prove a generalization of the central bounding theorem in [11] under significantly weaker assumptions. Aspects of this generalization are described in the proceedings paper [1] and the complete theorem is given in the journal article [2], which is currently under review. The key improvements to the theory fall into three categories:

1. *Invariant equations with nontrivial domains*. To prove the central bounding theorem in [11], it was necessary to assume that the operator $\mathscr{I}_g$ is well-defined for any interval argument. Using any conceivable approach for defining this operator, this requires that g itself is defined on all of $\mathbb{R}^{n_x}$. This is not problematic when g is affine, but is clearly limiting for nonlinear invariants since it prohibits any invariants involving divisions, square-roots, etc. In contrast, our new theory only requires the domain of g to be open, and therefore permits invariants described by much more general nonlinear functions.

2. *Parameter and input dependence*. The bounding theorem in [11] only permits invariants that depend exclusively on the system states. Specifically, the invariants cannot depend on uncertain model parameters or inputs. In contrast, our new theorem permits constraints with arbitrary dependence on both uncertain time-invariant parameters and uncertain time-varying inputs, as well as on the time-derivatives of the states.

3. *General state constraints*. Our new bounding theorem permits $\mathscr{I}_g$ to make refinements based on any state constraints of interest, rather than just on solution invariants. For example, nonlinear inequalities that hold for all solutions of (1) can now be used. Moreover, our new theory also permits refinements based on externally imposed state constraints that need not hold for all solutions of (1), provided that one is only interested in bounding the solutions that do satisfy these constraints. This capability is useful in algorithms for solving open-loop optimal control problems to guaranteed global optimality and potentially elsewhere [12].

In aggregate, the new contributions above significantly increased the applicability of our bounding algorithms, enabling us to address a wide variety of systems with invariants and other state constraints that are nonlinear and potentially dependent on uncertain model parameters, inputs, and state derivatives. One example of these new capabilities is provided after the discussion of algorithms for nonlinear invariants in §2.2.4. Several further examples are given after the discussion of manufacturing invariants for general nonlinear systems in §2.3.1.

2.2.4 An Efficient Differential Inequalities Bounding Algorithm Exploiting Nonlinear Invariants

An efficient new bounding algorithm was developed to implement the theory discussed in the previous section, resulting in the first method capable of making use of nonlinear invariants $g(x(t;x_0,w))=0$ (possible dependence of g on w and $\dot{x}$ is omitted for simplicity of exposition, although it is permissible in the developed theory and algorithm). As noted above, the key challenge in implementing the nonlinear theory was to develop an algorithm for the refinement operator $\mathscr{I}_g$. Given such an algorithm, the overall bounding method involves simply solving (7)–(8) with any state-of-the-art numerical integration code to obtain x^L and x^U.

The fundamental requirement on $\mathscr{I}_g$ is that it satisfies the inclusion $\mathscr{I}_g(Z) \supset \{z \in Z : g(z)=0\}$, $\forall Z \in \mathbb{IR}^{n_x}$. Although there is a large literature on bounding the solutions of nonlinear systems of equations in non-dynamic settings, there are two issues that make the application to reachability analysis uniquely challenging. First, details of DI theory require $\mathscr{I}_g$ to satisfy the Lipschitz condition (9), which is of no concern in standard uses of interval refinement methods. Second, solving the ODEs (7)–(8) requires executing $\mathscr{I}_g$ in every time step of the numerical integration, so the efficiency of $\mathscr{I}_g$ is critically important. After investigating several alternatives, we ultimately developed an effective algorithm for $\mathscr{I}_g$ based on a modified form of the interval Krawczyk method. The interval Krawczyk method is an interval Newton-type method based on the following consequence of the mean-value theorem:

$$z,c \in Z,\ g(z)=0 \qquad \Longrightarrow \qquad -g(c) \in \frac{\partial g}{\partial x}(Z)(z-c), \tag{14}$$

where $\frac{\partial g}{\partial x}(Z)$ is an interval enclosure of the Jacobian of g. Interval Newton methods use various rearrangements of this inclusion to obtain tighter bounds on each z_j implied by the constraint $g(z)=0$ [13]. The interval Krawczyk method is one of the weaker methods in this class, but is very efficient and always satisfies the Lipschitz condition (9), whereas other methods fail without restrictive assumptions on the form of g [14]. Our modified Krawczyk algorithm is given in detail in [1] and [2], and is rigorously proven to satisfy all of the requirements of our new DI theorem in [2]. Notably, this algorithm also benefits from the complexity reduction technique developed for affine invariants discussed in §2.2.2. Numerical case studies indicate that this algorithm is highly effective at reducing the conservatism of fast DI bounding methods on the basis of nonlinear invariants. One example is provided next and additional examples are given after the discussion of manufacturing invariants in §2.3.1.

Example 2. 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 [15, 16]. The ODEs are

$$\dot{x}(t) = u_1 x(t)(1-y(t)), \qquad \dot{y}(t) = u_2 y(t)(x(t)-1), \tag{15}$$

with horizon $[t_0,t_f]=[0,10]$ 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, parameter-dependent solution invariant, regardless of the values of u_1 and u_2:

$$u_2\big[\ln(x(t)/x_0)-(x(t)-x_0)\big]+u_1\big[\ln(y(t)/y_0)-(y(t)-y_0)\big]=0. \tag{16}$$

Figure 2 shows that exploiting the nonlinear invariant (16) using our new method leads to very sharp bounds on the solutions of (15), 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[1]. Our method also significantly outperformed other state-of-the-art bounding methods in the literature. For example, the bounds from our new method are much tighter than those obtained using the linear

[1] 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. Numerical integration was done using the Sundials solver CVODE [17] with absolute and relative tolerances of 10^{-5}.

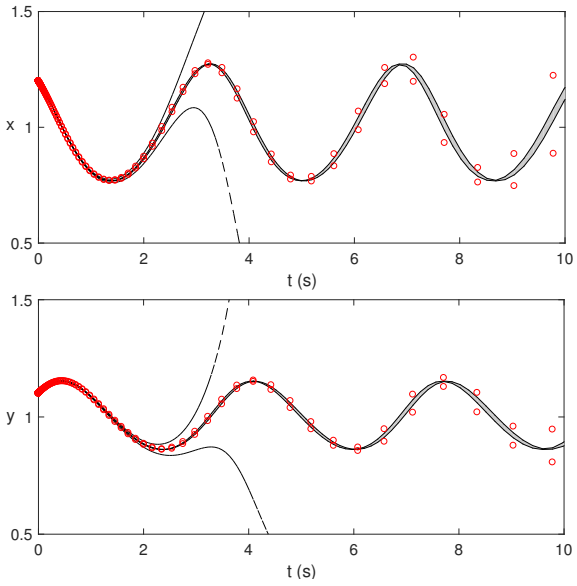

Figure 2: State bounds for x and y in (15) computed using SDI (dashed black) and our new method exploiting the nonlinear invariant (16) (red circles) with sampled solutions (gray shaded region).

programming-based polyhedral bounding algorithm in [15], which required 0.050s, and very similar to the bounds obtained using the Taylor Model algorithm in [16], which required 0.59s.

2.3 Research Findings from Task 2

The bounding methods developed in Task 1 only apply to models that naturally satisfy solution invariants. The objective of Task 2 was to extend these methods to general nonlinear systems. Our approach was to first develop a theoretical framework for introducing solution invariants into arbitrary dynamic models, thereby enabling the advanced bounding methods developed in Task 1 to be applied. Since this approach leaves considerable flexibility in the choice of the introduced invariants, we next aimed to develop broadly effective strategies for designing invariants that would be maximally effective at reducing the conservatism of the computed bounds. The results of this project on both of these issues are described in the subsections below and are discussed in further detail in the publications [1–3, 5, 6].

2.3.1 A New Method for Manufacturing Invariants by Lifting into a Higher-Dimensional State-Space

A theoretical framework was developed for introducing solution invariants into general nonlinear systems by lifting these systems into a higher-dimensional state space. This is the central contribution of this project because it enables highly effective reachable set bounding methods for systems with invariants to be applied to general nonlinear systems for the first time. This new approach, which is described in detail in [3], proceeds by first choosing a continuously differentiable function $g : \mathbb{R}^{n_x} \to \mathbb{R}^{n_y}$ and defining the new state variables $y(t;x_0,w) \equiv g(x(t;x_0,w))$. These definitions are then differentiated to form the *augmented system*

$$\frac{d}{dt}\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} f(t,x(t),w(t)) \\ \frac{\partial g}{\partial x}(x(t))f(t,x(t),w(t)) \end{bmatrix}, \quad \begin{bmatrix} x(t_0) = x_0 \\ y(t_0) = g(x_0) \end{bmatrix}. \tag{17}$$

Clearly, if (x,y) is a solution of (17), then x is a solution of (1). Thus, state bounds for (17) provide state bounds for (1). Moreover, by design, (17) implies that

$$\begin{aligned} y(t_0) - g(x(t_0)) &= 0 \quad \text{and} \\ \frac{d}{dt}[y(t) - g(x(t))] &= \dot{y}(t) - \frac{\partial g}{\partial x}(x(t))\dot{x}(t) = 0, \end{aligned} \tag{18}$$

which together imply that the solutions of (17) satisfy the invariants $y(t) - g(x(t)) = 0$, $\forall t \in [t_0, t_f]$ and (x_0, w). We call these invariants *manufactured invariants* to distinguish them from any invariants that may be satisfied by the solutions of the original model. State bounds for (17) can now be computed using the methods developed in Task 1. This procedure is summarized in the following theorem, which is the central result of our publication [3].

Theorem 1. *Choose any differentiable $g : \mathbb{R}^{n_x} \to \mathbb{R}^{n_y}$ such that $\frac{\partial g}{\partial x}$ is locally Lipschitz continuous and* (17) *has a unique solution on $[t_0, t_f]$ for every admissible (x_0, w). Let $\mathscr{I}_g$ be an interval refinement operator satisfying $\mathscr{I}_g(Z) \supset \{z \in Z : g(z) = 0\}$, $\forall Z \in \mathbb{IR}^{n_x}$, and the Lipschitz condition* (9)*. Moreover, let*

$$h(t,z,v) \equiv \frac{\partial g}{\partial x}(z) f(t,z,v), \qquad \forall (t,z,v) \in [t_0,t_f] \times \mathbb{R}^{n_x} \times W, \tag{19}$$

and let $[h_j^L, h_j^U]$, $[g_j^L, g_j^U]$, and $[f_j^L, f_j^U]$ be inclusion monotonic interval extensions of h_j, g_j, and f_j, respectively. Finally, let $x^L, x^U : [t_0,t_f] \to \mathbb{R}^{n_x}$ and $y^L, y^U : [t_0,t_f] \to \mathbb{R}^{n_y}$ be solutions of the following ODEs for all $i \in \{1,\ldots,n_x\}$ and $j \in \{1,\ldots,n_y\}$, where $Z(t) = \left[\begin{bmatrix} x^L(t) \\ y^L(t) \end{bmatrix}, \begin{bmatrix} x^U(t) \\ y^U(t) \end{bmatrix}\right]$:

$$\begin{aligned} \dot{x}_i^L(t) &= f_i^L(t, \mathscr{I}_g(\beta_i^L(Z(t))), W) & x_i^L(t_0) &= x_{0,i}^L \\ \dot{x}_i^U(t) &= f_i^U(t, \mathscr{I}_g(\beta_i^U(Z(t))), W) & x_i^U(t_0) &= x_{0,i}^U \\ \dot{y}_j^L(t) &= h_j^L(t, \mathscr{I}_g(\beta_{n_x+j}^L(Z(t))), W) & y_j^L(t_0) &= g_j^L(X_0) \\ \dot{y}_j^U(t) &= h_j^U(t, \mathscr{I}_g(\beta_{n_x+j}^U(Z(t))), W) & y_j^U(t_0) &= g_j^U(X_0) \end{aligned} \tag{20}$$

Then $x(t;x_0,w) \in [x^L(t), x^U(t)]$ and $y(t;x_0,w) \in [y^L(t), y^U(t)]$ for all $t \in [t_0,t_f]$ and all admissible (x_0,w).

Theorem 1 shows that valid state bounds for (1) can be computed by bounding the augmented system (17) rather than the original ODEs, and that the manufactured invariants g can be exploited in doing so. However, Theorem 1 does not ensure that this will result in improved bounds, and provides no guidance on how to choose effective g functions. Our experience with a large number of test problems over the course of the project shows that there are nearly always choices of g that produce bounds that are significantly sharper than those produced by applying standard DI to (1) [3]. Although effective choices of g can be difficult to identify, we have found that there are general strategies that tend to be effective for many models within the same physical domain. In the following two subsections, we discuss effective strategies for models composed of transient mass and energy balances, which are ubiquitous in chemical and biological engineering, and for models describing vehicle dynamics under path or trajectory tracking control. Numerical examples in both of these subsections clearly demonstrate the effectiveness of Theorem 1 when paired with an appropriate choice of g. Finally, subsection 2.3.4 describes a new bounding method called mean-value differential inequalities (MVDI) that can be interpreted as a fully automated method for choosing effective g functions. MVDI is applicable to general nonlinear systems but may be less efficient than the tailored methods discussed in subsections 2.3.2–2.3.3.

2.3.2 *Effective Manufactured Invariants for Mass and Energy Balance Models*

A general strategy has been developed for choosing effective invariants for dynamic models composed of transient mass and energy balances, which are ubiquitous in chemical and biomolecular engineering. This strategy exploits the fact that ODEs in this class often involve sums of nonlinear terms with the same or very similar terms appearing in multiple ODEs. Thus, the ODEs in (1) take the special form

$$\dot{x}(t) = h(t,x(t),w(t)) + Sr(t,x(t),w(t)), \tag{21}$$

where r is a vector function whose components describe the rates of various physical processes (chemical reactions, mass transfer between phases, heat transfer, etc.) and S is an $n_x \times n_r$ matrix describing the effect of each of these processes on each state variable. The function h contains terms that are not shared between multiple ODEs and is zero in the simplest cases, such as in batch reactor models. Even when h is nonzero, the terms that tend to make bounding difficult (due to strong nonlinearities, large uncertainties, or both) overwhelmingly appear in r, not in h. Accordingly, our basic strategy is to consider linear combinations of the original ODEs that result in complete or partial cancellation of these terms. Specifically, this is done by choosing an appropriate matrix $U \in \mathbb{R}^{n_y \times n_x}$, defining $y = Ux$, and forming the augmented system

$$\frac{d}{dt}\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} h(t,x(t),w(t)) + Sr(t,x(t),w(t)) \\ Uh(t,x(t),w(t)) + USr(t,x(t),w(t)) \end{bmatrix}, \quad \begin{bmatrix} x(t_0) = x_0 \\ y(t_0) = Ux_0 \end{bmatrix}. \tag{22}$$

At a minimum, we populate U with a maximal set of linearly independent rows u_i^{T} that satisfy $u_i^{\mathrm{T}} S = 0$. When this holds, sharp bounds are likely to be obtained for the corresponding y_i, which in turn provides useful information about the possible values of x through the relation $y_i = u_i^{\mathrm{T}} x$. Once these choices of u_i^{T} are exhausted (there may be none), further rows can be added to U such that $u_i^{\mathrm{T}} S$ has a single nonzero element. This produces states y_i that are only effected by a single component of r, and hence a single physical process, and these can often be bounded much more accurately than states that are affected by all processes. Note that if S contains uncertain parameters, then this strategy requires U to be uncertain as well. This leads to nonlinear invariants, but these can be readily handled by the methods developed in Task 1.

The articles [1, 3] provide many examples of the application of this strategy. Although it is clearly conceivable to automate a form of this strategy, we determined that pursuing automation would have consumed an undue fraction of the project resources. This decision was based largely on our judgment that our published examples already provide sufficient guidance for a competent practitioner to successfully apply the strategy by hand. This, along with the fact that this strategy is limited to a specific class of models, however important, led us to conclude that finding more general solutions and/or similar solutions for other classes of problems was likely to be more impactful.

A small selection of examples from [1,3] are given below. These results show that, when highly nonlinear and uncertain terms can be canceled, the developed strategy results in very significant improvements in bound accuracy and/or computational efficiency relative to other state-of-the-art approaches. Moreover, due to the mathematical similarities described above, this strategy is effective in surprisingly diverse systems, from metabolic reaction networks to industrial separation systems such as liquid-liquid extraction columns where no reactions occur at all.

Example 3. The following ODEs describe an anaerobic wastewater treatment process with pH self-regulation

and liquid-gas transfer from [18]:

$$\begin{aligned}
\dot{X}_1 &= (\mu_1(S_1) - \alpha D)X_1 \\
\dot{X}_2 &= (\mu_2(S_2) - \alpha D)X_2 \\
\dot{S}_1 &= D(S_1^{in} - S_1) - k_1\mu_1(S_1)X_1 \\
\dot{S}_2 &= D(S_2^{in} - S_2) + k_2\mu_1(S_1)X_1 - k_3\mu_2(S_2)X_2 \\
\dot{Z} &= D(Z^{in} - Z) \\
\dot{C} &= D(C^{in} - C) - q_{CO_2} + k_4\mu_1(S_1)X_1 + k_5\mu_2(S_2)X_2
\end{aligned} \tag{23}$$

where

$$\begin{aligned}
q_{CO_2} &= k_L a(C + S_2 - Z - K_H P_{CO_2}) \\
P_{CO_2} &= \frac{\phi_{CO_2} - \sqrt{\phi_{CO_2}^2 - 4K_H P_t (C + S_2 - Z)}}{2K_H} \\
\phi_{CO_2} &= C + S_2 - Z + K_H P_t + \frac{k_6}{k_L a}\mu_2(S_2)X_2 \\
\mu_1(S_1) &= \bar{\mu}_1 \frac{S_1}{S_1 + K_{S_1}} \\
\mu_2(S_2) &= \bar{\mu}_2 \frac{S_2}{S_2 + K_{S_2} + S_2^2/K_{I_2}}
\end{aligned} \tag{24}$$

The time horizon is $[t_0, t_f] = [0, 20]$ 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}, and $C(t_0) \in [39.2, 40.8]$ mmolL^{-1}, and the parameters $k_1 \in [42.14, 42.98]$ g(COD) g(cell)$^{-1}$ and $k_2 \in [116.5, 118.24]$ 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 as in [19].

To the best of our knowledge, (23) does not obey any existing solution invariants. Thus, we apply the strategy outlined in §2.3.1 to embed the model into a higher-dimensional augmented system that satisfies solution invariants by design. Specifically, we define the redundant state variables

$$N_1 \equiv k_1 X_1 + S_1, \quad N_2 \equiv -k_2 X_1 + k_3 X_2 + S_2, \tag{25}$$

and augment (23) with the corresponding ODEs for N_1 and N_2 derived by differentiating (25). After some simplification, these are

$$\begin{aligned}
\dot{N}_1 &= D(S_1^{in} + S_1(\alpha - 1) - \alpha N_1), \\
\dot{N}_2 &= D(S_2^{in} + S_2(\alpha - 1) - \alpha N_2).
\end{aligned} \tag{26}$$

As discussed above, the variables N_1 and N_2 are chosen such that highly nonlinear and uncertain terms cancel when deriving (26) from (23) (specifically, $\mu_1(S_1)X_1$ and $\mu_2(S_2)X_2$, which describe enzymatic reactions). By construction, the solutions of the lifted system consisting of (23) and (26) satisfy the nonlinear invariants

$$\begin{aligned}
0 &= -N_1 + k_1 X_1 + S_1, \\
0 &= -N_2 - k_2 X_1 + k_3 X_2 + S_2.
\end{aligned} \tag{27}$$

Figure 3 compares the standard DI method applied directly to (23) with our new method applied to the lifted system consisting of (23) and (26) with the nonlinear invariants (27). Again, standard DI produces

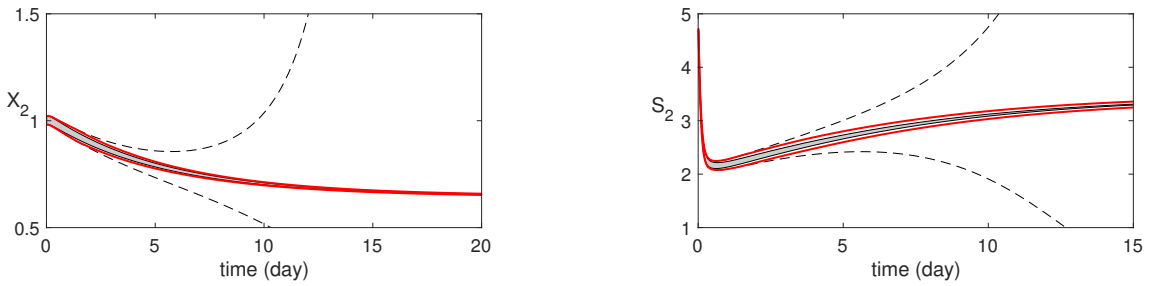

Figure 3: State bounds for X_2 and S_2 in (23) computed using SDI (dashed black) and our new method exploiting the nonlinear invariants (27) (solid red) with sampled solutions (gray shaded region).

rapidly diverging bounds. However, the use of the invariants (27) results in very sharp bounds over the entire time horizon, and appears to stabilize the bounds as $t \to \infty$. Our new method required 5.0×10^{-2}s to produce the bounds shown in Figure 3, compared to 7.0×10^{-3}s for standard DI. For reference, integrating a single trajectory of (23) required 2.8×10^{-4}s on average[2]. This problem was also considered in [19] over the shorter horizon $[t_0, t_f] = [0, 4]$ 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.

Example 4. The following model describes a two-phase counter-current multistage liquid-liquid extraction system with a single solute [20]:

$$\begin{aligned} V_L \dot{x}_n &= L(x_{n-1} - x_n) - Q_n \\ V_G \dot{y}_n &= G(y_{n+1} - y_n) + Q_n \end{aligned} \tag{28}$$

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

$$Q_n = K_L a(x_n - x_n^*)V. \tag{29}$$

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), 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 zero 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 (28), effective redundant state variables can be created by arranging for the cancellation of this term. Specifically, we define

$$N_n = V_L x_n + V_G y_n, \quad n = 1, \ldots, 5, \tag{30}$$

which leads to the augmented ODEs

$$\dot{N}_n = L x_{n-1} + G y_{n+1} - (L x_n + G y_n), \tag{31}$$

$$= 5(x_{n-1} + y_{n+1}) - \frac{5}{2} N_n, \quad n = 1, \ldots, 5. \tag{32}$$

[2]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. Numerical integration was done using the Sundials solver CVODE [17] with absolute and relative tolerances of 10^{-5}.

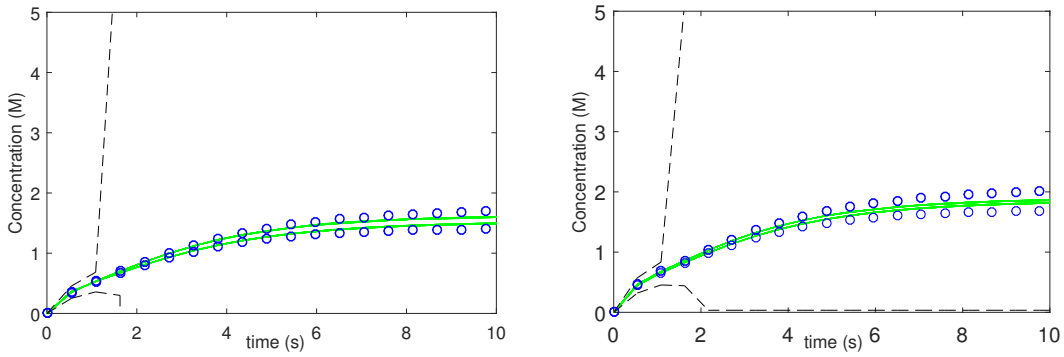

Figure 4: State bounds for x_5 (left) and y_5 (right) from (28) computed using standard DI (dashed) and and our new approach with manufactured invariants as in Theorem 1 (circles). Solid lines are real trajectories

Figure 4 shows that the bounds produced by applying the standard DI method to (28) rapidly diverge. In contrast, the use of manufactured invariants provides bounds that are nearly exact. The computational costs were 0.04s for a single trajectory and 0.7s for DI using manufactured invariants[3], while integration of the standard DI bounds failed due to rapid divergence around $t = 2$.

2.3.3 Effective Manufactured Invariants for Path and Trajectory Tracking Problems

Some preliminary new strategies have been developed for choosing effective invariants for dynamic models describing vehicles under path and trajectory tracking control. It is presently unclear how broadly effective these strategies are, but we have obtained very promising results for several test cases. These results are the subject of a new manuscript in preparation [4].

Models in this domain have clear mathematical similarities stemming from the basic equations of motion even when they describe very different vehicles. By considering several test cases, we identified two key challenges for bounding the solutions of this class of models. First, the governing ODE for each state typically consists of just a single nonlinear term, and the same term is not repeated in the ODEs for multiple states. Thus, the term-cancellation strategy outlined in §2.3.2 is almost entirely ineffective. In fact, all conceivable strategies for this class of problems lead to nonlinear invariants, which are much more difficult to automate. Second, the models in this domain are closed-loop models; i.e., the dynamics can be written as

$$\dot{x}(t) = h(t, x(t), w(t), \kappa(x(t))), \tag{33}$$

where h describes the open-loop vehicle dynamics and κ is a state-feedback law. Surprisingly, we found that this structure (which clearly arises in a much wider range of applications) is deeply problematic for standard DI bounding methods, as well as any other method relying heavily on interval arithmetic. This is because the feedback law causes a significant interval dependency problem in (33). In brief, this means that standard interval methods will treat the two occurrences of $x(t)$ in (33) (one in the open-loop dynamics and one in the feedback law) as independent for the purposes of bounding. Thus, the intended effect of the feedback law (i.e., to counteract and override the natural x-dependence of h) will not be reflected in the computed bounds. As a consequence, bounds computed for the closed-loop system can be much worse than those computed using a nominal open-loop input, despite the fact that the true reachable set is much smaller under feedback.

To address these issues, we obtained path and/or trajectory tracking controllers for a few basic vehicle models from the literature and studied the corresponding closed-loop models in detail. Based on these case studies, we identified the following two strategies as broadly effective for reducing the conservatism of fast DI bounding methods:

[3]Methods were implemented in MATLAB using the numerical integrator CVODE with default settings [17] on a Dell Precision T3610 with an Intel Xeon E5-1607 v2 processor @ 3.00 GHz.

1. *Apply the bounding method to the ODEs describing the vehicle dynamics in the same coordinate system that was used to derive the feedback law.* In trajectory tracking problems, the feedback law is nearly always derived by considering some kind of error dynamics. Importantly, when the closed-loop system is formed in the appropriate error coordinates, the action of the control law (i.e., the manner in which it counteracts and overrides the natural dynamics of the system) often manifests itself explicitly as term cancellations or other major algebraic simplifications that are not apparent in the original (x,y,z) workspace coordinates. These simplifications can have a profound impact on the accuracy of interval arithmetic, and hence on the accuracy of DI bounding methods. Thus, applying DI to the error dynamics (appropriately simplified) and subsequently transforming the results back to the workspace coordinates of interest is likely to result in much more accurate reachability bounds than applying DI directly in the workspace coordinates. In addition to considering error coordinates, path tracking controllers are often derived by considering a transformed independent variable representing some measure of progress along the path (e.g., arclength) rather than time. Forming the closed-loop error dynamics with respect to this transformed variable can result in further algebraic simplifications that are advantageous for interval arithmetic. Moreover, as demonstrated below, computing error bounds with respect to this transformed variable also makes it possible to convert the bounds back to the workspace coordinates of interest with much less conservatism.

2. *Define new variables y as Lyapunov-like functions for the closed-loop system.* Path and trajectory tracking controllers are commonly proven to be stable with the aid of a Lyapunov function $\mathscr{V}$ for the closed-loop error dynamics. Therefore, once a controller has been designed for a vehicle of interest, $\mathscr{V}$ is readily available. We have found that defining the new state variable $y = \mathscr{V}(x)$ and then applying the bounding method described in §2.3.1 often results in much tighter reachability bounds than standard DI. To see why, note that this choice of y leads to an augmented system of the form (17) where the ODE for $\dot{y}$ is exactly the total time derivative of $\mathscr{V}$ (i.e., its Lie derivative with respect to the vector field defined by the closed-loop error dynamics). By the definition of a Lyapunov function, the right-hand side of this ODE is negative in a neighborhood of the origin. Moreover, in order to prove this fact in practice, $\mathscr{V}$ is nearly always designed such that this right-hand side can be expressed in a simple algebraic form. Specifically, it benefits from exactly the kind of term cancellations and other simplifications that are highly beneficial for interval arithmetic. Thus, applying the bounding method described in §2.3.1 typically results in sharp bounds on $y(t) = \mathscr{V}(x(t))$ that can then be used to effectively refine the bounds on $x(t)$ pointwise in time as they are propagated forward.

Our manuscript in preparation [4] provides several examples of the application of these strategies. One of these examples is provided below showing that the combination of these two strategies can lead to sharp bounds over much longer time horizons than standard approaches.

Example 5. Consider the following vehicle dynamics with position (x,y) and heading angle θ:

$$\dot{x} = v\cos(\theta), \quad \dot{y} = v\sin(\theta), \quad \dot{\theta} = \omega. \tag{34}$$

The objective is to control the heading rate ω in order to track a reference path described by a smooth curve $(x_{ref}(\gamma), y_{ref}(\gamma))$ parameterized by its arclength γ and curvature $c(\gamma)$. We specify $c(\gamma) = 1/30$ for $\gamma \in [0,80]$ m and $c(\gamma) = -1/30$ for $\gamma \in (80,160]$ m.

Provided that the vehicle stays sufficiently close to the reference path, its location $(x(t), y(t))$ at any time t has a unique nearest point on the path, $(x^*(t), y^*(t))$ [21]. Let $s(t)$ be the arclength at $(x^*(t), y^*(t))$ and let $n(t)$ be the unit tangent to the path at $(x^*(t), y^*(t))$. Moreover, let $e_x(t) = x(t) - x^*(t)$ and $e_y(t) = y(t) - y^*(t)$ and define the tracking error by $e(t) = e_x(t)n_y(t) - e_y(t)n_x(t)$. Finally, let $\theta_e(t)$ be the difference between $\theta(t)$ and the reference heading angle $\theta_{ref}(t)$ defined by the tangent to the reference path at $(x^*(t), y^*(t))$. With

these definitions, the vehicle's motion can be described equivalently by the following error dynamics [21]:

$$\dot{s} = \frac{v\cos(\theta_e)}{1-c(s)e}, \quad \dot{e} = v\sin(\theta_e), \quad \dot{\theta}_e = \omega - \frac{vc(s)\cos(\theta_e)}{1-c(s)e}. \tag{35}$$

Based on this representation, the following path tracking controller is proposed [21]:

$$\omega = \frac{vc(s)\cos(\theta_e)}{1-c(s)e} - \left(2.8\sqrt{v^2+0.1}\right)\theta_e - \left(4v\frac{\sin(\theta_e)}{\theta_e}\right)e. \tag{36}$$

We are interested in computing bounds on the possible vehicle positions $(x(t), y(t))$ under this path-tracking controller starting from the uncertain initial conditions $(x_0, y_0, \theta_0) \in [0,0] \times [0.8,1] \times [\frac{\pi}{12}, \frac{\pi}{6}]$ and with uncertain time-varying velocity $v(t) \in [5,6]$ m/s. Perhaps the most natural approach to achieve this is to apply a bounding algorithm to the closed-loop system obtained by substituting (36) into (34). However, this cannot be implemented because the coordinate transformation between (x,y,θ) and (s,e,θ_e) (which would be required to evaluate the control law) cannot be written as a closed-form algebraic expression, and so there is no simple way to propagate interval bounds through it. However, even if this were possible (which it is for other examples), it would likely lead to very conservative bounds due to the dependency problem caused by feedback described in §2.3.3. Both problems are potentially solved by instead applying a bounding algorithm to the closed-loop error dynamics obtained by substituting (36) into (35), which results in a significant term cancellation that is beneficial for interval arithmetic. However, this only yields bounds on (s,e,θ_e), and again, there is no simple way to propagate these bounds back to the (x,y) coordinates of interest. To work around this problem, the bounding algorithm can instead be applied to the following system describing the closed-loop error and original dynamics simultaneously:

$$\begin{aligned} &\dot{s} = \frac{v\cos(\theta_e)}{1-c(s)e}, \quad \dot{e} = v\sin(\theta_e), \quad \dot{\theta}_e = -\left(2.8\sqrt{v^2+0.1}\right)\theta_e - \left(4v\frac{\sin(\theta_e)}{\theta_e}\right)e, \\ &\dot{x} = v\cos(\theta_d+\theta_e), \quad \dot{y} = v\sin(\theta_d+\theta_e), \quad \theta_d = \frac{vc(s)\cos(\theta_e)}{1-c(s)e}. \end{aligned} \tag{37}$$

The results of applying standard DI to this system are shown in red in Figure 5. Unfortunately, the results are very weak despite taking advantage of a term cancellation in the error dynamics.

To make a further improvement, we next define a new state variable based on the following Lyapunov function for the closed-loop error dynamics given in [21]: $\mathscr{V}(e,\theta_e) = \frac{1}{2}(e^2 + \frac{1}{4}\theta_e^2)$. Defining $V(t) = \mathscr{V}(e(t),\theta_e(t))$ (we use V instead of y for the augmented variable to avoid conflict with the vertical position y), we obtain the additional ODE (after simplification)

$$\dot{V} = -\frac{2.8\sqrt{v^2+0.1}}{4}\theta_e^2. \tag{38}$$

By design, the augmented system comprising (37) and (38) satisfies the invariant

$$V = \frac{1}{2}\left(e^2 + \frac{1}{4}\theta_e^2\right). \tag{39}$$

The result of applying the DI method for systems with nonlinear invariants developed in Task 1 to the system comprising (37) and (38) with invariant (39) is shown in purple in Figure 5. These bounds constitute a significant improvement over standard DI, but are still quite weak. However, close examination of these results shows that only the bounds on the original variables (x,y,θ) are weak, while the computed bounds on (s,e,θ_e) are reasonably tight and much better than SDI (not shown). This suggests that much tighter bounds might be achievable through a better approach for mapping error bounds back into the coordinates of interest.

After some experimentation, we discovered that choosing the arclength s instead of t as the independent variable in path tracking problems makes it significantly easier to propagate interval bounds from (e,θ_e) to (x,y) without significant additional conservatism. This is possible because, when the vehicle remains sufficiently close to the path, (37) shows that $\dot{s}$ is non-negative. Therefore, s is a monotonically increasing function of t and is a valid change of variables. Executing this change of variables for the error states and the Lyapunov function V gives

$$\frac{de}{ds} = (1-c(s)e)\tan\theta_e, \quad \frac{d\theta_e}{ds} = (1-c(s)e)\left(\frac{-\left(2.8\sqrt{v^2+0.1}\right)\theta_e}{v\cos\theta_e} - \frac{4\tan\theta_e}{\theta_e}e\right), \tag{40}$$

$$\frac{dV}{ds} = -(1-c(s)e)\frac{\left(2.8\sqrt{v^2+0.1}\right)\theta_e^2}{4v\cos(\theta_e)}.$$

Applying a bounding algorithm to (40) gives bounds on (e,θ_e) as functions of s. Since each s values refers to specific location along the reference path and $e(s)$ is essentially a signed distance between the vehicles location and the reference path at s in the direction orthogonal to the path at s, bounds on the original positions can be directly obtained by

$$x^{L/U} = x_{ref} + e^{L/U}\cos\left(\theta_{ref}+\frac{\pi}{2}\right) \quad \text{and} \quad y^{L/U} = y_{ref} + e^{L/U}\sin\left(\theta_{ref}+\frac{\pi}{2}\right). \tag{41}$$

The result of applying standard DI to the system (40) and subsequently propagating the bounds to the (x,y) space via (41) is shown in blue in Figure 5, while the result of applying the advanced DI method for systems with nonlinear invariants developed in Task 1 to (40) with invariant (39) is shown in green. It can be seen that bounding in this coordinate system provides some advantages for SDI relative to the case where the independent variable is t, but the bounds still rapidly diverge. In contrast, the combination of using the correct coordinate systems and exploiting the Lyapunov invariant (39) through our new DI method leads to very sharp bounds over a much longer horizon than any other method. Moreover, this method required only 0.016 s of CPU time[4].

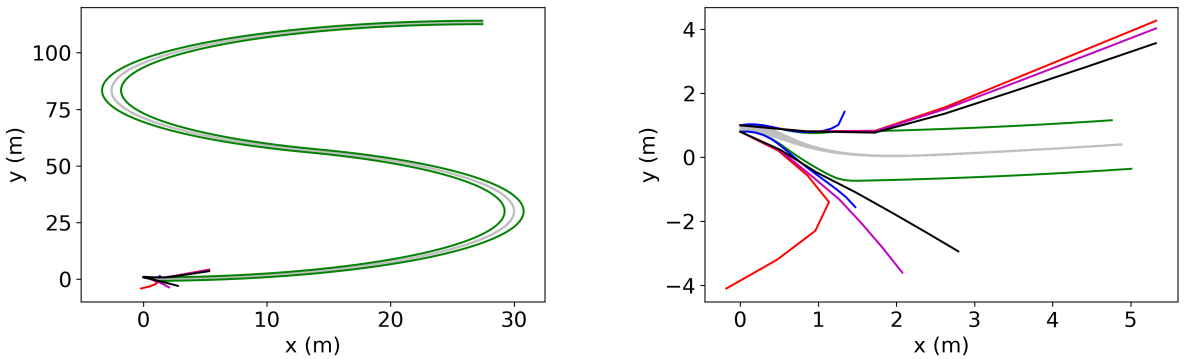

Figure 5: Bounds on the vehicle position for Example 5 produced by applying standard DI to (37) (red), applying advanced DI to the system comprising (37) and (38) with invariant (39) (purple), applying standard DI to (40) (blue), and applying advanced DI to (40) with invariant (39) (green). The black bounds are not discussed in this report. Sampled trajectories are grey.

[4]C++ implementation on a laptop with a 2.9 GHz Intel Core i5 and ODEs solved using the SUNDIALS solver CVODE [17] with absolute and relative tolerances of 10^{-5}.

2.3.4 Mean-Value Differential Inequalities for General Nonlinear Models

A new bounding method called mean-value differential inequalities (MVDI) has been developed that can be interpreted as a fully automated method for choosing effective invariants. This method is described in detail in our publications [5,6]. MVDI is applicable to general nonlinear systems but may be less efficient than the tailored methods discussed in §2.3.2–2.3.3. The central idea of this technique is to automatically construct approximate invariants through the use of the forward sensitivity equations for the model of interest, and then to derive a rigorous interval refinement algorithm based on these approximate invariants through an application of the Mean Value Theorem. To present this technique, we first restrict our attention to ODEs affected by time-invariant uncertainties p:

$$\dot{x}(t) = f(t,x(t),p), \quad x(t_0) = x_0(p), \quad t \in [t_0,t_f], \quad x(t) \in \mathbb{R}^{n_x}, \quad p \in \mathbb{R}^{n_p}, \tag{42}$$

with $p \in P$ and $x_0 : P \to \mathbb{R}^{n_x}$ for some interval uncertainty set $P = [p^L, p^U]$. Recall that the sensitivities for (42) are defined as

$$s_{ij}(t,p) \equiv \frac{\partial x_i}{\partial p_j}(t,p), \quad i \in \{1,\ldots,n_x\}, \quad j \in \{1,\ldots,n_p\}. \tag{43}$$

Let s denote the matrix with components s_{ij} and define the functions

$$f_s(t,x,s,p) \equiv \frac{\partial f}{\partial x}(t,x,p)s + \frac{\partial f}{\partial p}(t,x,p), \qquad s_0(p) \equiv \frac{\partial x_0}{\partial p}(p). \tag{44}$$

With these definitions, the joint state and sensitivity vector $\left[\begin{smallmatrix} x \\ s \end{smallmatrix}\right]$ is well-known to satisfy the following initial value problem in $n_x + n_x n_p$ ODEs:

$$\frac{d}{dt}\begin{bmatrix} x(t,p) \\ s(t,p) \end{bmatrix} = \begin{bmatrix} f(t,x(t,p),p) \\ f_s(t,x(t,p),s(t,p),p) \end{bmatrix}, \qquad \begin{bmatrix} x(t_0,p) \\ s(t_0,p) \end{bmatrix} = \begin{bmatrix} x_0(p) \\ s_0(p) \end{bmatrix}. \tag{45}$$

In our new approach, we choose the sensitivities s as the new state variables, i.e. $y = s$, so that (45) becomes the augmented system (17). At this point, the development of our new method deviates slightly from our previous methods because s has not been defined as an explicit algebraic function of x as in $s = g(x)$, and so (45) does not seem to satisfy any invariants by design. However, a relation between x and s that can be used for bound refinement is provided in this case by the Mean Value Theorem (MVT). Specifically, given any $i \in \{1,\ldots,n_x\}$ and any reference point $\hat{p} \in P$, the MVT ensures that there exists $\xi(t,p) \in P$ such that

$$x_i(t,p) = x_i(t,\hat{p}) + s_i^{\mathrm{T}}(t,\xi(t,p))(p-\hat{p}), \qquad \forall (t,p) \in [t_0,t_f] \times P, \tag{46}$$

where s_i^{T} denotes the i^{th} row of s. Note that this equation relates $x(t,p)$ to $s(t,\xi(t,p))$, whereas a true invariant would relate $x(t,p)$ to $s(t,p)$. However, we have shown that, because $\xi(t,p) \in P$, this relation can still be used to make valid refinements to interval bounds on $x(t,p)$ and $s(t,p)$ computed using differential inequalities. Thus, in the context of the reachability algorithms developed in Task 1, Equation (46) can be treated as an invariant for the augmented system (45), and valid bounds on x can be computed by applying our differential inequalities approach to (45) while exploiting the nonlinear 'invariant' (46) through a suitably defined interval refinement algorithm $\mathscr{I}_g$. Critically, both the augmented system (45) and its invariants (46) are automatically generated in this scheme without the need for any user insights.

Because (46) is not a true invariant, its use is not strictly permitted by the theory developed in Task 1. Instead, the validity of this method is established by a more general theoretical development that this is main result of our proceedings paper [5] and a journal article in review [6].

Although we have done limited numerical experiments with this approach so far, the results have been very promising. In the examples below, we show that this approach is competitive with using manually

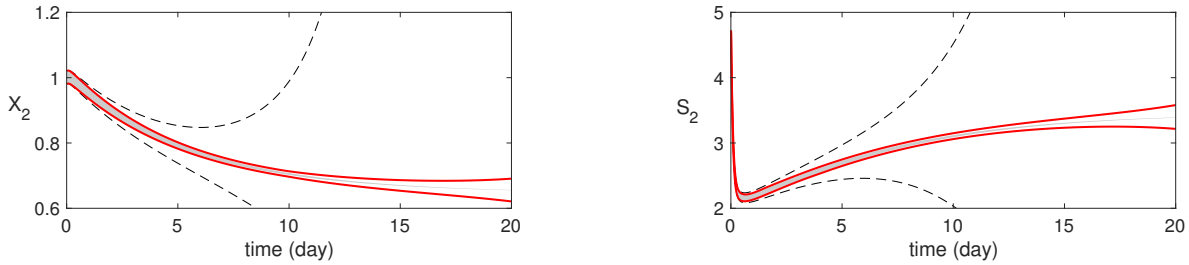

Figure 6: Bounds on X_2 and S_2 in (23) from standard DI (dashed black) and mean value DI (solid red) with sampled solutions (shaded gray area).

designed invariants for Example 3 above, and produces greatly improved bounds for a challenging aircraft trajectory tracking problem where no manually designed invariants have yet been effective.

Example 6. Consider again the ODEs in (23) with time horizon $[t_0, t_f] = [0, 20]$ days and uncertain 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}, and $C(t_0) \in [39.2, 40.8]$ mmolL^{-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 [19].

Figure 6 compares the standard differential inequalities method (SDI) to our new mean value DI method (MVDI) described above. We use the reference point $\hat{p} = \text{mid}(P)$ and the reference trajectory $x(t, \hat{p})$. Figure 6 clearly shows that SDI produces rapidly diverging bounds, while MVDI produces very sharp bounds. Moreover, the bounds in Figure 6 are only slightly weaker than those in Figure 3, which were computed with custom, manually constructed invariants. The time required for integrating a single trajectory of (23) is 4×10^{-4}s on average, while SDI takes 2.5×10^{-3}s and MVDI takes 2.6×10^{-2}s[5]. Thus MVDI is roughly a factor of 10 slower than SDI. This problem was also considered in [19] over the shorter horizon $[t_0, t_f] = [0, 4]$. 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 mean value differential inequalities provides sharp bound at significantly lower cost in this case.

Example 7. The following equations describe the motion of a fixed-wing UAV [22], where (x, y, z) is the UAV position, v_{xy} and v_z are the velocities in the $xy-$plane and $z-$plane, respectively, ψ is the heading angle, and θ is the roll angle:

$$\begin{aligned} &\dot{x} = v_{xy}\cos(\psi), \quad \dot{y} = v_{xy}\sin(\psi), \quad \dot{z} = v_z \\ &\dot{\psi} = \frac{g}{v_{xy}}\tan(\theta), \quad \dot{v}_{xy} = a_{xy}, \quad \dot{v}_z = a_z, \quad \dot{\theta} = \omega. \end{aligned} \tag{47}$$

The initial position is assumed to be uncertain but bounded in $X_0 \times Y_0 \times Z_0 = [-1, 1] \times [-1, 1] \times [-1, 1]$ m^3. The remaining initial conditions are $\psi = 0$ rad, $v_{xy} = 10$ m/s, $v_z = 1$ m/s, and $\theta = 0.3$ rad.

There are three control inputs, (a_{xy}, a_z, ω), where a_{xy} and a_z are the acceleration in the $xy-$plane and $z-$plane in m/s^2, respectively, and ω is the roll angle rate in rad/s. These inputs are used to track a desired trajectory, which is denoted by $(x_d(t), y_d(t), z_d(t), \psi_d(t), v_{xy,d}(t), v_{z,d}(t))$. Here, the desired trajectory has been generated by specifying the open-loop control input $(a_{xy}, a_z, \omega) = (1, 0.1, 0)$ and simulating the model without uncertainty from the initial position at the center of $X_0 \times Y_0 \times Z_0$. In order to track this trajectory

[5]C++ implementations on a laptop with a 2.9 GHz Intel Core i7 and ODEs solved using the SUNDIALS solver CVODE [17] with default settings.

under uncertainty, the following feedback control law is used

$$a_{xy} = k_5\varepsilon_x + k_6(v_{xy,d} - v_{xy}), \tag{48}$$
$$a_z = k_7\varepsilon_z + k_8(v_{z,d} - v_z), \tag{49}$$
$$\omega = k_4(k_1\varepsilon_y + k_2(\psi_d - \psi) + k_3(\dot{\psi}_d - \dot{\psi}) - \theta), \tag{50}$$

where ε_x, ε_y, and ε_z are the lateral, longitudinal, and altitude errors defined by

$$\varepsilon_x = \quad \cos(\psi_d)(x_d - x) + \sin(\psi_d)(y_d - y), \tag{51}$$
$$\varepsilon_y = -\sin(\psi_d)(x_d - x) + \cos(\psi_d)(y_d - y), \tag{52}$$
$$\varepsilon_z = z_d - z. \tag{53}$$

The gains are $k_1 = 0.05$, $k_2 = 5.0$, $k_3 = 5.0$, $k_4 = 1.0$, $k_5 = 0.1$, $k_6 = 1.0$, $k_7 = 0.13$, and $k_8 = 1.0$.

Figure 7 compares the standard differential inequalities method (SDI) to our new mean value DI method (MVDI) described above. The figure clearly shows that SDI (black) produces rapidly diverging bounds while MVDI (red) is able to produce bounds very close to the set of true trajectories shown in green. The time required for integrating a single trajectory of (47) is 3.6×10^{-4}s on average, while SDI takes 8.0×10^{-3}s and MVDI takes 2.8×10^{-2}s[6]. Thus, MVDI produces accurate bounds over 10s of flight time nearly 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 2.3×10^{-2}s), which is unlikely to provide a reliable approximation of the full reachable set.

We also computed bounds using the state-of-the-art reachability code CORA, which is based on the propagation of high-order zonotopes rather than intervals [23, 24]. The bottom panel in Figure 7 shows that CORA produces even tighter bounds than MVDI. However, CORA requires 3.6 s[7]. Thus, CORA is more than 125 times slower than MVDI and is unlikely to be fast enough for real-time verification and control.

2.4 Progress Towards Task 3

The objective of Task 3 was to develop algorithms and software for fast and accurate state bounding through the automatic identification, introduction, and exploitation of model redundancy. The specific subtasks originally proposed to accomplish this were largely predicated on the expectation that Task 2 would yield general strategies for designing invariants that would benefit greatly from automation, and that automating these strategies would require certain symbolic computing capabilities. However, this did not occur. Although Task 2 did indeed produce general strategies for designing effective invariants in two important application domains (mass and energy balance models and trajectory/path tracking problems), we determined that automating these strategies would not be the most impactful use of the remaining project resources for reasons described in §2.3.2–2.3.3. More importantly, work in Task 2 led to the development of the mean-value differential inequalities (MVDI) method, which automates the construction of effective invariants for general nonlinear systems without the need for advanced symbolic computing capabilities. Thus, the objective of Task 3 was accomplished through the development of the MVDI algorithm already discussed in §2.3.4 and the originally proposed subtasks involving advanced symbolic computing capabilities were not pursued further.

[6] C++ implementations on a laptop with a 2.9 GHz Intel Core i7 and ODEs solved using the SUNDIALS solver CVODE [17] with default settings.

[7] CORA is implemented in MATLAB and was run with zonotope order 7, time step 0.1 s, and all other parameters at their default settings on a laptop with a 2.5 GHz Intel Core i5.

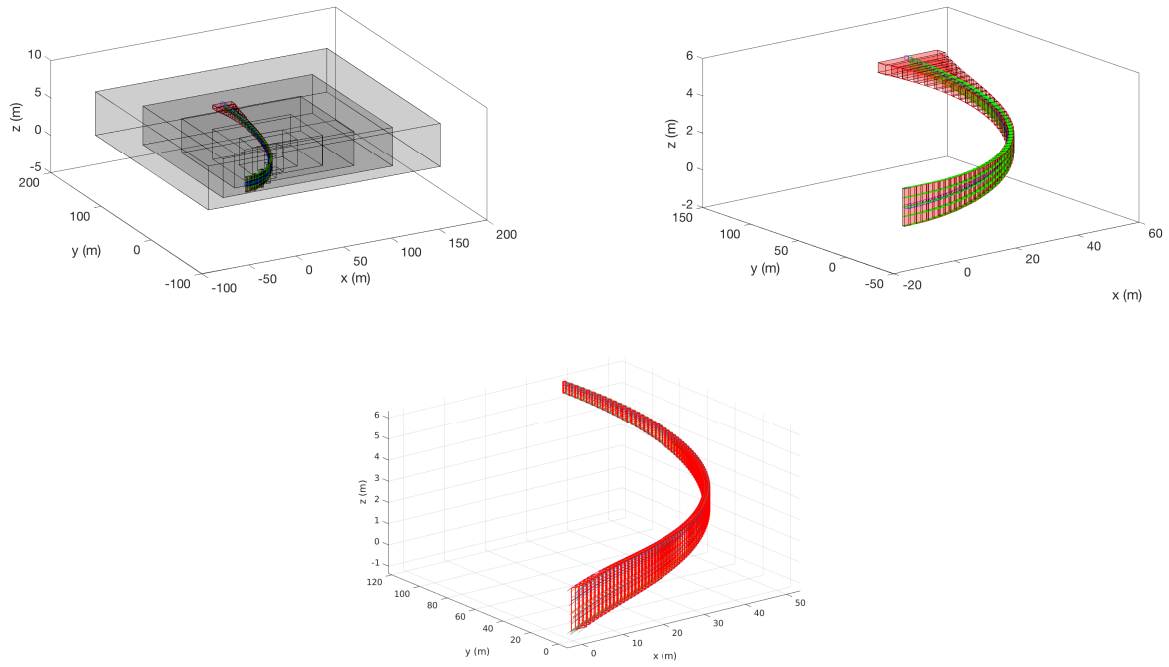

Figure 7: Top Left: Bounds on x, y, and z in (47) from standard DI (black boxes) and mean value DI (red boxes) with sampled solutions (green). Top Right: Same image with standard DI bounds removed for clarity. Bottom: Bounds computed by the state-of-the-art zonotope-based reachability code CORA (red boxes).

2.4.1 A Fast Interval Arithmetic Library for MATLAB

Although the bounding algorithms developed in this project do not require advanced symbolic computing, they do make extensive use of symbolic-numeric capabilities such as automatic differentiation (AD) and interval arithmetic (IA). Existing AD and IA packages were readily available for use in `C++` implementations of our bounding algorithms. However, existing tools for use in MATLAB implementations were not nearly efficient enough to support our use case. To address this, we developed a `MATLAB` library called `SymbComp` that automatically constructs computational graphs of nonlinear functions and supports very fast interval computations using a novel code generation technique. Using this technique, `SymbComp` is able to perform interval computations considerably faster that existing `MATLAB` libraries and competing libraries that we previously developed using operator overloading, which is the standard approach used to design interval libraries in many programming languages. The times required for `SymbComp` to execute some basic interval operations are compared with those from the commercial library `Intlab` and our own operator overloading implementation in Table 1. These results indicate that `SymbComp` is nearly an order of magnitude more efficient on average. The current version of `SymbComp` also supports advanced bounding computations with polyhedral outer approximations and more general convex enclosures. `SymbComp` is the subject of a journal manuscript currently in preparation and is expected to be released on GitHub sometime over the next year.

2.5 Other Findings

Through our interactions with AFOSR and AFRL personnel over the course of this project, we became aware of several problems of interest to the Air Force that are closely related to this project but not strictly included in the original proposal. In two cases, we were able to make significant progress on these problems under this award through reasonably straightforward extensions of the methods discussed in the previous

Table 1: Interval arithmetic operation times in seconds per hundred runs in MATLAB 2014b with the JIT compiler disabled on a computer with 4 Intel® Core™ i5-3210M cores at 2.5GHz and 4 Gigabytes of RAM.

Library	$x+y$	$x \times y$	x^y	$\log x$
Symbcomp	0.017035	0.040372	0.038328	0.021660
Operator Overloading	0.118257	0.140825	0.139114	0.103205
Intlab	0.094486	0.198328	0.344083	0.398991

sections. These are outlined in the following two subsections.

2.5.1 Rapid and Accurate Uncertainty Propagation for Discrete-Time Systems

The primary thrust of this project was to develop improved reachable set bounding methods for nonlinear dynamic systems described by ordinary differential equations in continuous-time. However, many robust estimation and control problems of potential interest to AFOSR are commonly formulated in discrete-time and require reachability analysis capabilities in this setting. One example is the calculation of optimal open-loop control inputs that must satisfy state constraints robustly under uncertainty. Such calculations are commonly used in moving horizon predictive control schemes formulated in discrete time. After studying the current state-of-the-art in discrete-time reachability analysis, we identified a clear opportunity to make a significant contribution by extending our continuous-time methods to this setting.

Notably, there was previously no direct analogue of the standard differential inequalities (DI) approach in discrete time, so our new advances in DI could not be directly applied. Conceptually, this is because DI theory depends critically on the fact that, for continuous-time systems, a trajectory cannot leave a set X without crossing its boundary. Thus, to propagate a reachable set enclosure forward in time, it suffices to consider the behavior of the vector field on its boundary [25]. This leads to the use of the operators $\beta_i^{L/U}$ in the bounding system (5)–(6). Unfortunately, this is not true in discrete-time, and this precludes any straightforward analogue of the DI approach for general discrete-time systems.

In practice, however, discrete-time systems are often obtained by Euler discretization of continuous-time models. Focusing on this special case, our main new results show that, for any given system, there exists a bound on the discretization step size below which a discrete-time analogue of the basic DI method provides valid bounds on the reachable sets of the discretized system. This step size bound can be easily computed in advance, and is no more restrictive than the step size required to preserve basic physical properties of the solution, such as non-negativity of certain states. Additionally, we showed that the advanced DI methods using invariants developed in this project [3] are also valid in discrete time under a slightly tighter step size restriction. These results are the subject of a proceedings paper [7] and a journal article currently under review [8]. Our numerical results in both publications show that our new discrete-time DI methods substantially outperform the existing state-of-the-art discrete-time methods [26, 27] (which compute bounding sets in the form of high-order zonotopes) for some challenging test cases.

2.5.2 Formal Verification Using Rapid and Accurate Backward Reachability Analysis

We developed a new method for solving a class of formal verification problems that can be formulated as backward reachability problems for systems described by nonlinear ODEs. This is the subject of a new paper in preparation [9].

In contrast to forward reachability analysis, which aims to compute an enclosure of the set of trajectories reachable from a given set of uncertain inputs and/or initial conditions, backward reachability analysis aims to characterize the set of uncertain inputs that generate trajectories that satisfy a given set of conditions. We specifically considered conditions that can be written in terms of a target set that must be reached eventually

and a collection of obstacle sets that must be avoided at all times. Many important problems can be written in this form, including aircraft collision avoidance problems, chemical process safety verification, etc. Existing backward reachability algorithms for general nonlinear systems fall into two broad categories. The first involves the solution of a Hamilton-Jacobi-Bellman PDE, which scales exponentially in the number of states and uncertain parameters and is therefore intractable for many systems of interest. The second applies a recursive sequence of forward reachability computations within a branch-and-bound (B&B) algorithm. More specifically, the uncertainty set is adaptively partitioned until most subsets can be proven by forward reachability analysis to be either safe or unsafe, with the remaining sets accounting for a total volume less than some prescribed tolerance. A significant advantage of this approach is that it does not scale exponentially in the state dimension (unless a forward reachability subroutine is used that does). Moreover, although it does scale worst-case exponentially in the number of uncertain parameters, its performance in practice is dictated by the speed and accuracy of the embedded forward reachability calculations. Unfortunately, existing forward reachability algorithms are either too conservative or too inefficient to make this procedure effective beyond simple test cases [28].

To address this problem, we developed a branch-and-bound-based backward reachability algorithm based around the advanced forward reachability methods developed under this award (see §2.2–2.4). Although this algorithm has only been tested on a small number of examples to date, our preliminary results show that the improved speed and accuracy of our forward reachability algorithms enables the solution of some verification problems orders of magnitude faster than the current state of the art. Moreover, these methods make verification possible for significantly larger systems (in terms of the state dimension in particular) than would be tractable through alternative approaches. Two representative examples are given below.

Example 8. The following ODEs describe the exothermic reaction $A \rightarrow B$ in a batch reactor fitted with a segmented, variable-area cooling jacket:

$$\begin{aligned} \dot{X} &= k_0 e^{-E_a/RT}(1-X), \\ \dot{T} &= \frac{UA}{C_{A0}VC_p}(T_a - T) - \frac{\Delta H_R}{C_p} k_0 e^{-E_a/RT}(1-X). \end{aligned} \tag{54}$$

Above, X is the reaction conversion and T is the reactor temperature. The final time is $t_f = 1500$ s. The initial temperature is uncertain and satisfies $T_0 \in [310, 540]$ K. Moreover, there are two control variables, the coolant temperature T_a and the heat transfer coefficient UA, which must obey $T_a \in [290, 310]$ K and $UA \in [2.5, 3.5]$ W/K. All other parameters are constant at the values in [28].

The formal verification problem of interest is defined in terms of the following four discrete states associated with (54) at each point in time t:

- s_1 (safe operation): $t < t_f$ and the reactor temperature has not exceeded 540 K for any $t' \in [t_0, t]$;
- s_2 (unsafe operation): $t \leq t_f$ and the reactor temperature has exceeded 540 K at some $t' \in [t_0, t]$;
- s_3: (safe and on-spec run): $t = t_f$, the reactor temperature has not exceeded 540 K for any $t' \in [t_0, t_f]$, and $X(t_f) \geq 0.975$;
- s_4 (safe but off-spec run): $t = t_f$, the reactor temperature has not exceeded 540 K for any $t' \in [t_0, t_f]$, but $X(t_f) < 0.975$;

The system begins in state s_1 and immediately transitions to s_2, s_3, or s_4 when the required conditions are met. All trajectories end in s_2, s_3, or s_4. The backward reachability problem of interest is to characterize the subsets in the joint space of uncertainties $T_0 \in [310, 540]$ K and control actions $T_a \in [290, 310]$ K and $UA \in [2.5, 3.5]$ W/K that lead to trajectories ending in states s_2, s_3, and s_4.

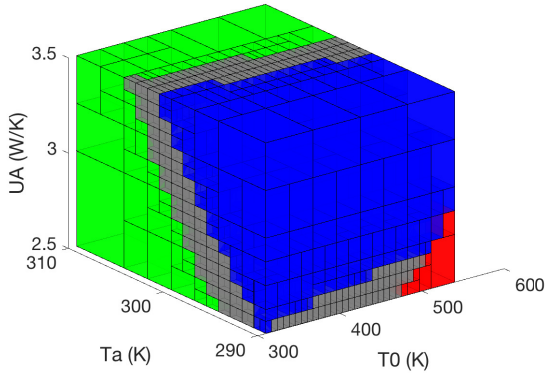

Figure 8: Guaranteed inner-approximations of the backward reachable sets for discrete states s_2 (unsafe, red), s_3 (safe and on-spec, green), and s_4 (safe but off-spec, blue) computed using DI with manufactured invariants with an undecided volume threshold of 20%. The undecided region is shown in gray.

This problem was originally posed in [28] and solved using a branch-and-bound backward reachability algorithm. In each iteration of this algorithm, the required forward reachability calculation was done using a method based on sets described by high-order Taylor models. To compare, we applied two similar branch-and-bound algorithms. In the first, the forward reachability computations are done using the standard differential inequalities (DI) method. In the second, they are done using the advanced differential inequalities method with manufactured invariants described in §2.3.1. To apply the latter, we defined the new state variable

$$N = \frac{\Delta H_R}{C_p} X + T \tag{55}$$

and augmented (54) with the redundant ODE

$$\dot{N} = \frac{UA}{C_{A0} V C_p} (T_a + \frac{\Delta H_R}{C_p} X - N). \tag{56}$$

Figure 8 shows the backward reachable sets for states s_2, s_3, and s_4 computed by DI with manufactured invariants (i.e., the subsets of the joint uncertainty and control input space that generate trajectories ending in discrete states s_2, s_3, and s_4). In this case, the algorithm was set to terminate when the volume of the set that remains undecided is less than 20% of the original volume. Using DI with manufactured invariants, this occurred after the joint uncertainty and control input space was partitioned into $3,348$ subintervals at a total cost of 2.8 CPUs[8]. In contrast, standard DI required $4,545$ subintervals and 3.48 CPUs to achieve the same tolerance. Thus, the use of invariants results in a 20% reduction in CPU time.

To compare with the prior state-of-the-art method in [28], this computation was repeated with an undecided volume tolerance of 4.8% in order to match the accuracy of the solution in [28]. Using DI with manufactured invariants, the algorithm terminated after the joint uncertainty and control input space was partitioned into $58,383$ subintervals at a total cost of 49.6 CPUs. In contrast, the Taylor model method in [28] required $48,440$ subintervals and $98,649$ CPUs. Thus, the advanced DI algorithm with manufactured invariants developed under this award enables the solution of this formal verification problem well over three orders of magnitude faster than the prior state of the art.

[8] C++ implementations on a laptop with a 2.9 GHz Intel Core i7 and ODEs solved using the SUNDIALS solver CVODE [17] with absolute and relative tolerances of 10^{-6}.

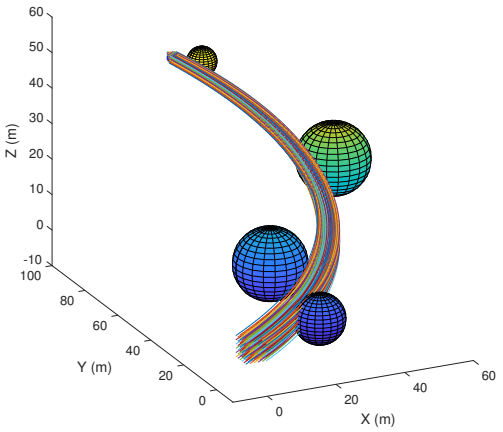

Figure 9: Obstacles and sampled trajectories for Example 9.

Example 9. Consider again the fixed-wing UAV model described in Example 7. Using the same reference trajectory and tracking controller as in Example 7, the backward reachability problem of interest here is to compute the set of initial conditions from which the UAV is guaranteed to avoid collision with four obstacles placed around the reference trajectory. The obstacles are all spheres with their centers placed at the (x,y,z)-coordinates $(25,0,5)$, $(20,30,10)$, $(48,50,28)$, and $(25,82,48)$ and with radii 7, 10, 10, and 4, respectively. The uncertain initial location of the UAV satisfies $x_0 \in [-3,3]$ m, $y_0 \in [-3,3]$ m, and $z_0 \in [-3,3]$ m. All other initial conditions are fixed as in Example 7 and the time horizon is $[t_0,t_f] = [0,0.8]$ s. The four obstacles are shown with several sampled trajectories in Figure 9.

To solve this problem, we again applied a branch-and-bound-based backward reachability algorithm as described in §2.5.2. For the forward reachability calculations required in each iteration, we considered both standard DI and the mean-value DI (MVDI) algorithm developed under this award and described in §2.3.4. Figure 10 shows the backward reachable sets computed by MVDI with a 20% undecided volume threshold. Due to the accuracy of the forward reachable sets produced by MVDI, this computation required only 0.57 CPUs[9] and partitioned the uncertainty space into only 28 subintervals. In contrast, SDI requires 142.86 CPUs and 107,796 subintervals. With a tighter undecided volume threshold of 5%, MVDI requires 5.7 CPUs and 308 subintervals, while SDI requires 185 CPUs and 132,987 subintervals. Thus, the MVDI method developed under this award enables this verification problem to be solved much more efficiently than using previously available methods.

3 Supported Personnel

Faculty:	Joseph K. Scott, Assistant Professor
Graduate Students:	Dr. Kai Shen; PhD in Chemical Engineering; March 2019
	Ms. Xuejiao Yang*; PhD in Chemical Engineering; Expected May 2021
	Mr. Dillard Robertson†; PhD in Chemical Engineering; Expected May 2024
	Mr. Taehun Kim†; PhD in Chemical Engineering; Expected May 2023
	Mr. Yuanxun Shao†; PhD in Chemical Engineering; Expected May 2022

* Partially supported with AFOSR funds

† Supported for 1 month or less to assist with project close-out and preparation of the final report.

[9] C++ implementations on a laptop with a 2.9 GHz Intel Core i7 and ODEs solved using the SUNDIALS solver CVODE [17] with absolute and relative tolerances of 10^{-6}.

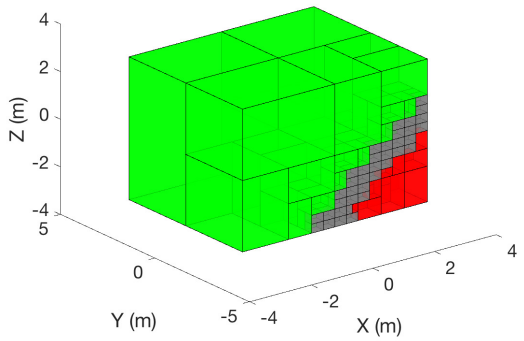

Figure 10: Guaranteed inner-approximations of the sets of initial conditions that are guaranteed to avoid all collisions (green) and guaranteed to lead to a collision (red) computed by MVDI with an undecided volume threshold of 20%. The undecided region is shown in gray.

4 Publications Supported by this Project

Journal Publications:

1. Shen, K. and Scott, J.K., "Rapid and Accurate Reachability Analysis for Nonlinear Dynamic Systems by Exploiting Model Redundancy," Computers and Chemical Engineering, **106**, 596–608, 2017

Refereed Conference Proceedings:

1. Shen, K. and Scott, J.K., "Tight Reachability Bounds for Nonlinear Systems using Nonlinear and Uncertain Solution Invariants," Proceedings of the 2018 American Control Conference, 2018
2. Yang, X. and Scott, J.K., "Efficient Reachability Bounds for Discrete-Time Nonlinear Systems by Extending the Continuous-Time Theory of Differential Inequalities," Proceedings of the 2018 American Control Conference, 2018
3. Shen, K. and Scott, J.K., "Mean Value Form Enclosures for Nonlinear Reachability Analysis," Proceedings of the 2018 IEEE Conference on Decision and Control, 2018

Journal Publications Under Review:

1. Shen, K. and Scott, J.K., "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, 2018
2. Yang, X. and Scott, J.K., "Accurate Uncertainty Propagation for Discrete-Time Nonlinear Systems Using Differential Inequalities with Model Redundancy," IEEE Transactions on Automatic Control, 2018
3. Shen, K. and Scott, J.K., "Tight Reachability Bounds for Constrained Nonlinear Systems Using Mean Value Differential Inequalities," Automatica, 2019

Journal Publications in Preparation:

1. Shen, K. and Scott, J.K., "Faster Solution of Backward Reachability Problems for Nonlinear Systems using Advanced Forward Reachability Algorithms," 2019
2. Yang, X. and Scott, J.K., "Guaranteed Safe Path and Trajectory Tracking via Reachability Analysis Using Differential Inequalities," 2019

5 Interactions/Transitions Supported by this Project

Other interactions include the following technical presentations:

1. Scott, J.K., "Rapid and Accurate Reachability Analysis for Nonlinear Systems by Exploiting Model Redundancy," Kolchin Seminar in Differential Algebra, New York University (March 2019)

2. Scott, J.K., "Rapid and Accurate Uncertainty Propagation, Safety Verification, and Fault Detection in Chemical, Aerospace, and Robotic Systems," Department of Chemical and Biological Engineering, University of Wisconsin-Madison (February 2019)

3. Scott, J.K., "Algorithms for Guaranteed Safety Verification and Fault Detection in Chemical, Aerospace, and Robotic Systems," Department of Chemical and Biomolecular Engineering, Georgia Institute of Technology (January 2019)

4. Shen, K., Yang, X., and Scott, J.K., "Verifying Performance Specifications for Dynamic Processes Under Uncertainty Using Backward Reachability Analysis," Computing and Systems Technology Division Plenary, Annual Meeting of the American Institute of Chemical Engineers (AIChE), Pittsburg, PA (October 2018)

5. Scott, J.K., "Dealing with Large Uncertainties in the Simulation and Optimization of Complex Chemical and Energy Systems," School of Chemical and Biomolecular Engineering, Georgia Institute of Technology (March 2018)

6. Shen, K. and Scott, J.K., "Improved Bounds on the Solutions of Nonlinear Dynamic Systems Using Centered-Form Differential-Inequalities," Annual Meeting of the American Institute of Chemical Engineers (AIChE), Minneapolis, MN (October 2017)

7. Shen, K. and Scott, J.K., "Rapid and Accurate Uncertainty Propagation for Nonlinear ODEs using Nonlinear Solution Invariants," Annual Meeting of the American Institute of Chemical Engineers (AIChE), San Francisco, CA (November 2016)

References

[1] Kai Shen and Joseph K. Scott. Tight reachability bounds for nonlinear systems using nonlinear and uncertain solution invariants. In *Proc. American Control Conference*, 2018.

[2] Shen K. and J. 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)*, 2018.

[3] Kai Shen and Joseph K. Scott. Rapid and accurate reachability analysis for nonlinear dynamic systems by exploiting model redundancy. *Computers & Chemical Engineering*, 106:596–608, 2017.

[4] X. Yang and J.K. Scott. Guaranteed safe path and trajectory tracking via reachability analysis using differential inequalities. *In Preparation*, 2019.

[5] Kai Shen and Joseph K. Scott. Mean value form enclosures for nonlinear reachability analysis. In *Proc. 57th IEEE Conference on Decision and Control*, 2018.

[6] K. Shen and J. K. Scott. Tight reachability bounds for constrained nonlinear systems using mean value differential inequalities. *Submitted*, 2019.

[7] Xuejiao Yang and Joseph K. Scott. Efficient reachability bounds for discrete-time nonlinear systems by extending the continuous-time theory of differential inequalities. In *Proc. American Control Conference*, 2018.

[8] X. Yang and J. K. Scott. Accurate uncertainty propagation for discrete-time nonlinear systems using differential inequalities with model redundancy. *Submitted*, 2018.

[9] K. Shen and J. K. Scott. Faster solution of backward reachability problems for nonlinear systems using advanced forward reachability algorithms. *In Preparation*, 2019.

[10] 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.

[11] J. K. Scott and P. I. Barton. Bounds on the reachable sets of nonlinear control systems. *Automatica*, 49:93–100, 2013.

[12] 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.

[13] A. Neumaier. *Interval Methods for Systems of Equations*. Cambridge University Press, Cambridge, 1990.

[14] J.K. Scott and P.I. Barton. Interval Bounds on the Solutions of Semi-Explicit Index-One DAEs. Part 2: Computation. *Numerische Mathematik*, 125(1):27–60, 2011.

[15] Stuart M. Harwood and Paul I. Barton. Efficient polyhedral enclosures for the reachable set of nonlinear control systems. *Mathematics of Control, Signals, and Systems*, 28(1):8, 2016.

[16] Y. Lin and M.A. Stadtherr. Validated Solutions of initial value problems for parametric ODEs. *Applied Numerical Mathematics*, 57:1145–1162, 2007.

[17] 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 Transactions on Mathematical Software*, 31:363–396, 2005.

[18] 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.

[19] Mario E. Villanueva, Boris Houska, and Benoît Chachuat. Unified framework for the propagation of continuous-time enclosures for parametric nonlinear ODEs. *Journal of Global Optimization*, 62(3):575–613, 2015.

[20] 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.

[21] Claude Samson. Path following and time-varying feedback stabilization of a wheeled mobile robot. In *Proceedings of the International Conference on Advanced Robotics and Computer Vision*, volume 13, pages 1–14, 1992.

[22] 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.

[23] M. Althoff. An introduction to CORA 2015. In *Proc. of the Workshop on Applied Verification for Continuous and Hybrid Systems*, pages 120–151, 2015.

[24] M. Althoff, O. Stursberg, and M. Buss. Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization. In *Proc. 47th IEEE Conference on Decision and Control*, pages 4042–4048, 2008.

[25] W. Walter. *Differential and Integral Inequalities*. Springer-Verlag, New York, 1970.

[26] C. Combastel. A state bounding observer for uncertain non-linear continuous-time systems based on zonotopes. In *Proc. 44th IEEE Conference on Decision and Control*, pages 7228–7234, 2005.

[27] T. Alamo, J.M. Bravo, and E.F. Camacho. Guaranteed state estimation by zonotopes. *Automatica*, 41(6):1035–1043, 2005.

[28] Youdong Lin and Mark A. Stadtherr. Rigorous model-based safety analysis for nonlinear continuous-time systems. *Computers & Chemical Engineering*, 33(2):493–502, 2009.